arXiv:2303.11445v1 [cs.FL] 20 Mar 2023

在 Isabelle/HOL 中形式化无限词与态射语言

Štěpán Starosta[2][0000−0001−5962−4297]

布拉格捷克理工大学，捷克共和国

摘要。 我们在通用证明助理 Isabelle/HOL 中，对无限词相关的基础内容进行了形式化。此外，我们还对纯态射语言和态射语言进行了形式化。最后，我们将斯特姆词形式化地定义为下机械词，并证明了一些非常基础的事实。这项形式化工作基于一个正在进行的大型词组合学形式化项目。

关键词： 无限词 · 态射语言 · 斯特姆词 · Isabelle/HOL · 数学形式化。

1 引言

无限词及其各种性质是词组合学中广受研究的对象。此类研究通常被认为始于 M. Prouhet 的[18] 1851年的工作以及 A. Thue [20] 约50年后的工作，而系统研究的开端以及这一领域的兴起，则以 Lothaire 第一部著作的问世为标志 [14]。关于其历史和所研究概念的详细介绍，可参阅 [3]。

本文从词组合学的角度，对与无限词相关的基础概念进行了形式化。数学形式化旨在以形式化方式写出命题，并使用给定的一组推理规则为其提供经机器验证的证明。随着计算能力的提升，自20世纪70年代四色定理首次通过（大规模）计算机辅助证明以来，这一领域取得了长足发展。读者可参阅综述 [12]，更详细地了解数学形式化的现状。从另一个角度来看，也可以通过各种现有证明器中100个重要定理的形式化现状 [21]，追踪这一领域及其可用工具的发展。

本文所介绍的形式化工作是在通用证明助手 Isabelle/HOL[11,17] 中完成的。Isabelle 是一个开源项目，其主发行版和形式化证明档案[1]中收录了大量形式化结果。Isabelle 的接口语言称为 Isar，参见 [16]。Isar 的目标之一是提供人类可读的证明。下文将给出许多从本文所介绍的形式化工作中原样复制的 Isar代码示例，至于其可读性如何，则留待读者评判。

该形式化工作是 Isabelle/HOL 中一个现有的词组合学形式化项目的一部分 [5]，该项目名为“形式化的词组合学”。截至目前，该项目涵盖了该领域的许多基础概念和一些进阶概念（与有限词相关）。除最新的代码库外[5]，，读者还可以参阅 [8,6,10]，了解该项目的更多详情。在此背景下，本文介绍了该项目在无限词概念方面的扩展，并在下文重点介绍此次扩展中重要或有趣的部分，省略了许多其他已形式化的论断和证明。

下一节将进一步介绍“形式化的词组合学”项目中的一些相关概念。该形式化工作分为 3 个部分，详见 第 3 至 5节。最后一节介绍形式化工作的结构。

2 预备知识——有限词的组合学

“形式化的词组合学”项目的基本组成部分名为“词组合学基础”，也收录于形式化证明档案 [9]，，其中简称为“词组合学_”。我们将使用这个名称来指代它，因为它是本文所介绍概念的形式化基础。出于同样的原因，本节将简要概述“词组合学_”中涉及的相关概念。

有限词是由给定集合 $\mathcal{A}$ 中的元素构成的有限序列，该集合通常称为字母表。字母表中的元素称为字母。有限词自然地表示为列表数据类型，在 Isabelle 中记作 'a list，其中 'a 是一个类型变量。因此，'a 表示字母的类型，而该类型的所有元素共同构成字母表。空列表/词记作 `[]` 或 Nil，在“词组合学_”中也记作 ε 。

连接操作也由 Isabelle 主发行版中的“List”理论提供。理论是 Isabelle 中形式化的基本单元，对应于扩展名为“thy”的文件。Combinatorics_Words 软件包定义了词组合学中更常用的连接记法，即在形式化中用通常的代数符号表示

。有限词的长度也在 List 理论中定义，并且采用了常见的记法 $|w|$[1]，Combinatorics_Words 中也采用类似的记法。同样，前缀和因子的概念已包含在理论“HOL-Library.Sublist” 中，Combinatorics_Words 会重用该理论。下面引入表示这些概念的常用记法：$p \leq_p w$ 表示 p 是 w的前缀，而 $f \leq_p w$ 表示 f 是 w的因子。上述记法以及许多相关的基础命题和略为进阶的命题，均收录于 Combinatorics_Words 的 “CoWBasic” 理论中。

端同态和态射的概念是在组合学_词中的“态射”理论里引入的，其中还包含许多有用的结论。态射 φ是从一个有限词集合到另一个有限词集合的映射，满足

[1]在形式化中，符号“|”以粗体显示，以避免与现有记号冲突。

对于所有 w, v，都有 $\varphi(w \cdot v) = \varphi(w) \cdot \varphi(v)$（它是从一个词幺半群到另一个词幺半群的同态）。

3 语言

对无限词的研究通常包括对其因子集合的研究。因此，在介绍无限词之前，我们自然会先介绍几个与语言相关的概念，也就是一般意义上的有限词集合。

在形式化中，语言无需另行定义，因为字母表是通过一种类型表示的。换句话说，语言是类型为 'a list set 的变量，即由类型为 'a list 的变量组成的集合，也就是有限词的集合。

首先定义对取因子运算封闭的语言。这种语言称为 因子闭合的：语言 L 是因子闭合的，当且仅当

$$\forall w \in L, \forall f, f \leq_f w \Rightarrow f \in L.$$

这个定义的形式化很直接：

定义 factorial :: "'a list set ⇒ bool"
其中 "factorial L ≡ (∀w f. w ∈ L ∧ f ≤f w ⟶ f ∈ L)"

类型 "'a list set ⇒ bool" 表示从 'a list 的集合（即有限词的集合）到 bool 的映射，因此构成了一个关于语言的谓词。符号 ⟶ 表示 HOL 蕴含，即作用于 bool 类型上的二元运算。

对语言取因子闭包 会 L 得到包含该语言的最小超集，且该超集是因子闭合的。在形式化过程中，使用命令**归纳集** ¯通过归纳定义闭包：

归纳集 factor_closure:: "'a list set ⇒ 'a list set" **对于** L ¯**其中** "w ∈ L ⟹ v ≤ f w ⟹ v ∈ factor_closure L"

长双箭头 $\Longrightarrow$ 是 Isabelle 用于推理命题的元蕴含。最后一行应读作：如果 $w \in L$ 且 $v \leq_f w$，那么 v 属于 factor_closure L。自由变量 w 和 v 都是全称量化的。使用这种非常简单的归纳定义，其优点是 Isabelle 会生成许多有用的事实，可在形式化过程中后续使用。例如，会生成以下名为 factor_closure.induct 的归纳规则：

```
"x ∈ 因子_闭包 L ⟹
(⋀w v. w ∈ L ⟹ v ≤f w ⟹ P v) ⟹ P x"
```

这条自动生成的规则可以理解为： x 属于语言 L 的因子闭包，并且对于所有 w, v，我们都有

$$w \in L, v \leq_f w \text{ implies that } P(v) \text{ holds},$$

那么 $P(x)$ 成立。

使用命令 **inductive_set** 还有其他好处。该命令用于定义以下形态语言概念，其思想源自

D0L 系统。D0L 系统是 L 系统的一种特例，该系统由 [13] 引入，用于模拟生物体的生长，如今则用于在虚拟世界中生成植被。相关概念的经典综述见 [19]。D0L 系统是一个三元组 $(\mathcal{A}, \varphi, w)$，其中 $\mathcal{A}$ 是一个字母表，φ 是 $\mathcal{A}^*$ 上的端同态，即 $\mathcal{A}$ 上所有有限词构成的集合，而 w 是 $\mathcal{A}$ 上的一个词，通常称为公理。集合 $\{\varphi^k(w) \mid k = 0, 1, \dots\}$ 是该系统的语言。我们关注该语言的因子闭包，也就是说，我们关注通过对公理 w 反复应用 φ 所能产生的任意因子。我们将这种语言称为纯形态语言，其定义如下：

归纳_集 纯_态射_语言:: "('a list ⇒ 'a list)⇒ 'a list ⇒ 'a list set"**对于** f 公理**其中**"公理 ∈ 纯_态射_语言 f 公理"| "w ∈ 纯_态射_语言 f 公理 ⟹ f w ∈纯_态射_语言 f 公理"| "w ∈ 纯_态射_语言 f 公理 ⟹ u ≤ f w ⟹ u ∈纯_态射_语言 f 公理"

上述纯_态射_语言的定义中有两个参数。第一个参数记作 f，类型为 'a list ⇒ 'a list，与端同态的类型相同，但这里并未要求它实际上是端态射。原因在于，定义本身不需要这一假设；稍后在证明关于这个归纳集的合理命题时，才会加入该假设。第二个参数是公理，记作 axiom。

有 3 条归纳规则：

1. 公理属于该语言；2. 该语言中任意元素经 f 映射后的像属于该语言；3. 该语言中任意元素的因子也属于该语言。

将底层数据类型视为字母表时，它与某个端同态 f 所对应的 D0L 系统的语言的因子闭包相同。另一方面，对于特定的公理选择，它与某个纯形态无限词的所有因子构成的集合相同，因此得名。若存在端同态 φ，使得 φ(u) = u 且 $\lim_{k\to+\infty} |\varphi^k(a)| = +\infty$，其中 a 是 u 的首字母，则称无限词 u 是纯形态的 。换言之，$\varphi^k(a)$构成了词 u 越来越长的前缀。

通过命令 **context** 将参数 f 固定为端同态。该命令引用组合学理论 “Morphisms” 中所需的局部语境 endomorphism。局部语境是 Isabelle 用来固定参数的工具，可避免在每个断言中重复这些参数。在此局部语境中，端同态变量固定为 f，而 $\mathcal{L}$ w 是 purely_morphic_language f w 的简写。

context 端同态**begin**

定义 bounded :: "'a list ⇒ bool"**其中** "
boundedw ≡ (finite (ℒ w))"

该定义表示，词 w 是有界的，当且仅当由它诱导的纯形态语言是有限的。当纯形态语言中有无限多个有界元素时，有界词便很有意义。这类系统称为 [4] *pushy*，我们也采用这一术语（f 仍固定为一个端同态）：

定义pushy :: "'a list ⇒ bool"**其中** "pushy 公理 ≡ ¬ 有限 ((ℒ 公理) ∩ (Collect bounded))"

作为一个形式化初等结论的亮点，我们选择以下命题：对于固定的端同态 f，公理是 pushy，当且仅当对于 f 的任意非平凡幂，该公理也是 pushy：

引理 pushy_strong_shift:**假设** "finite (UNIV::'a set)"**结论** "pushy 公理 = endomorphism.pushy(f^^(p + 1)) 公理"

上一条引理的假设指出，表示字母表的数据类型 'a 的全集是有限的。

对于非擦除端同态 f，证明相当简单；但如果不作此假设，证明就需要更加详尽。良好的形式化不仅要求作者用给定的形式语言正确写出每一项断言，还要求作者（一次又一次地）从多种等价的公式表述中，选择最适合后续复用的一种，因为通常大多数断言之后都会在进一步的形式化中再次使用。这种选择伴随着对假设的修订，往往最终会删去和／或推广这些假设。在这里，删去 f 为非擦除端同态这一假设所付出的代价，是证明变得稍微复杂一些，大致可按如下方式进行：

证明（**引理** pushy_strong_shift的证明概要）。从右到左的蕴含实际上很容易。我们证明其逆命题，即假设公理相对于 f 是 pushy。

首先假设字母像的最大长度大于 1。令Ψ = f^^(p + 1)。反设公理相对于 Ψ 不是 pushy。由此可知，存在一个常数 ℓ，使得Ψ^i(公理)的任何有界因子长度都小于 ℓ。由于公理相对于 f 是 pushy，因此存在一个足够长的有界因子，其原像是某个 Ψ^i(公理)，且长度必须大于 ℓ。这里，足够长是根据 f 的字母像最大长度确定的。由于有界因子的原像仍然有界，故得出矛盾。

字母像的最大长度不大于 1 时，剩下的情况很容易处理。

请注意，关于字母表有限性的剩余假设不可省略。事实上，考虑下面这个专门构造来使该结论不成立的例子。假设 $p > 0$，并考虑由以下内容确定的态射 φ

$$b_i \mapsto \begin{cases} b_{i+1}^{i+1} & \text{if } i+1 \neq k(p+1); \\ \varepsilon & \text{otherwise,} \end{cases} \qquad c_i \mapsto \begin{cases} c_{i+1} & \text{if } i \neq k(p+1); \\ b_{i+1}^{(i+1)!} c_{i+1} & \text{otherwise,} \end{cases}$$

若公理等于 c_0，则该断言不成立，因为我们得到

$$\varphi^i(c_0) = \begin{cases} c_i & \text{if } i = k(p+1); \\ b_i^{i!} c_i & \text{otherwise.} \end{cases}$$

由于 $\varphi^{p+1}(b_i) = \varepsilon$，每个字母 b_i 都是有界的，因此公理 c_0 相对于 φ是 pushy 的。另一方面，字母 c_i 并非有界（相对于φ，这等价于[2] 相对于 φ^{p+1}）。由于 $\left(\varphi^{p+1}\right)^i(c_0) = c_{i(p+1)}$，公理 c_0 相对于 φ^{p+1}不是 pushy。

态射 词 **v** 是一个纯态射词在态射 Ψ下的态射像，即 $\Psi(\varphi^k(\{\text{v20'>}))$ 构成词 **v** 越来越长的前缀，条件是 $\lim_{k\to+\infty} |\Psi(\varphi^k(a))| = +\infty$。因此，我们定义形态语言如下：**定义** 形态_语言 :: "('a list ⇒ 'a list) ⇒ ('a list⇒ 'b list) ⇒ 'a list ⇒ 'b list set"

其中 "形态_语言 f h 公理 ≡ 因子_闭包 (h‘纯_形态_语言 f 公理)"

变量 h 表示外层态射，而该语言就是 h 对纯_形态_语言 f 公理的像的因子闭包。也可以采用归纳定义，但这样就无法直接使用关于底层纯形态语言的事实。由于形式化时通常更倾向于重用已有事实，因此选择了直接但能力较弱的定义，并在形式化中明确证明了特定的有用结论。其中一个结论是以下固定端同态 f 和态射 h 的归纳规则：

引理 形态_语言_归纳[消耗 1]: **假设**"x ∈ 形态_语言 f h 公理" "P (h 公理)"**且** ih2: "⋀w. h w ∈ 形态_语言 f h 公理 ⟹ P (h w) ⟹P (h (f w))"**且**ih3: "⋀w u. w ∈ 形态_语言 f h 公理 ⟹ P w ⟹ u ≤f w ⟹ P u"**推出**"P x"

4 无限词

与有限词的数据类型的自然选择类似，无限词可以直接表示为类型 'a stream，该类型定义在理论 “HOL-

[2] 这一断言以 bounded_pow_bounded 之名进行了形式化。

Isabelle 主发行版中的 Library.Stream”。在这一理论中，类型 'a stream 被定义为如下的（余递归）余归纳数据类型：

余归纳数据类型 (sset: 'a) stream =
SCons (shd: 'a) (stl: "'a stream") (**右结合中缀** ‹##› 65)

该类型通过两个析构子定义。第一个是 shd，代表“流首”，类型为 'a，表示无限序列的第一个元素；第二个是 stl，代表“流尾”，类型为 'a stream，表示移除头部后的序列其余部分。这个有用的定义所做的远不止声明这两个析构子。例如，它还定义了构造子 SCons，记作 ##，将两个析构子联系起来：a ## uu 是一个无限序列，其头部等于 a，尾部等于 uu。

在词组合学的语境中，以下原始递归函数（来自“HOL-Library.Stream”）很有用，它将一个有限词前置到一个无限词上：

原始递归 shift :: "'a list ⇒ 'a stream ⇒ 'a stream" **其中**"shift [] s = s"| "shift (x # xs) s = x ## shift xs s"

它在 'a list 的两个构造子上定义：空词 []，以及 x # xs，后者表示将字母 a 前置到词 xs 上（这等同于将单元素词 [a] 与词 xs 连接）。在形式化中，我们采用符号 · 表示映射 shift。请注意，这是一个粗体的 cdot 符号，与表示两个有限词连接的非粗体符号不同。尽管 Isabelle 允许对符号进行重载，我们仍作此选择。原因是重载在某些情况下会阻止自动类型推断，导致经常需要显式指定变量的数据类型，进而使断言不够易读。

无限词的前缀和因子以一种直接明了的方式定义：**定义** sprefix:: "'a list ⇒ 'a stream ⇒ 布尔值"
其中 "sprefix p uu ≡ (∃vv. uu = p·vv)"

定义 sfactor:: "'a list ⇒ 'a stream ⇒ bool"
其中 "sfactor w uu ≡ (∃p vv. uu = p·w·vv)"

由于“prefix”和“factor”已分别用于有限词的前缀和因子，因此这里的前缀采用字母 s 表示，这与库“HOL- Library.Stream”中的用法一致（hd 是有限词的首字母，tl 是其尾部）。

无限词的语言就是其所有因子的集合。

定义 词_语言 ("$\mathcal{F}$")
其中 "词_语言 uu ≡ {w. sfactor w uu}"

我们采用记号 $\mathcal{F}$uu 表示无限词uu 的语言。同样，无限词的因子复杂度的定义也很直接，即统计给定长度的因子数量的映射：

定义 因子_复杂度**为** "因子_复杂度 uu ≡ (λi. 基数 (($\mathcal{F}$ uu)ᵢ))"

这里，Li 表示语言 L 中长度为 i 的元素。术语基数表示集合的基数。符号 λ 引入一个 λ 表达式，使因子 _ 复杂度成为从自然数到自然数的映射。

到目前为止，将流 'a 用于词组合学的改编是直接明了的。接下来的概念则没那么显而易见。首先，我们想根据一个表达式生成一个无限词，该表达式给出其第 n 个字母。这可以通过以下原始余递归函数实现：

原始余递归 snth_stream :: "(nat ⇒ 'a) ⇒ 'a 流" **其中**"shd (snth_streams) = (s 0)"| "stl (snth_streams) = snth_stream (λi. s (Suc i))"

snth_stream s 的参数 s 的类型为 nat ⇒ 'a，也就是说，它是一个从自然数到字母表的映射，表示该无限词的字母序列。这个余递归函数根据无限词的两个析构子来定义：首先说明其头部为 s 0，其尾部则通过将 s 平移 1 来定义，因为 Suc i 是 i 的后继，也就是 i+1。

另一种常用的生成新无限词的方法，是将一个态射应用于某个无限词。这可以通过组合"HOL-Library.Stream"和初等字组合学软件包中的现有工具来实现：

定义 morph_stream::"('a list ⇒ 'b list) ⇒ 'a stream ⇒ 'b stream"

其中 "morph_stream f uu ≡ flat (sfilter (λw. w ≠ ε) (smap f$^{\mathcal{C}}$uu))"

映射 f 的类型为 a list ⇒ 'b list。事实上，出于与上文讨论纯形态语言时相同的原因，这一定义并不要求 f 是一个态射。项 f$^{\mathcal{C}}$ 表示 f 的"核心"，即将字母（而非词）映射到其对应图像的底层映射。映射 smap 将 f$^{\mathcal{C}}$ 应用于 uu 的每个元素，从而生成一个词序列，更确切地说，是一个类型为 'b list list 的项。要将这个词序列连接成所需的类型为 'b list 的词，只需使用映射 flat。不过，展平后的序列会先经过筛选：利用 sfilter 函数忽略序列中的空词。之所以要进行这一步，是因为当展平后的序列中含有空词时，flat 的定义无法正常工作。

因此，给定的定义所得结果正是我们通常所说的对无限词 u $= u_0u_1u_2\ldots$ 的映射：

$$\varphi(\mathbf{u}) = \varphi(u_0)\varphi(u_1)\varphi(u_2)\ldots$$

唯一略有不同的情况是 φ(u) 实际上可能是一个有限词。在这种情况下，我们最终会得到一个项 morph_stream f uu，对它无法证明任何合理的命题。

引入 morph_stream 的动机之一是定义态射的不动点，而这通常需要一些假设，以防止将无限词缩减为有限词的情况。因此，上述定义是令人满意的。

更具体地说，这一假设是指态射 φ 在其不动点的首字母上可延拓，也就是说，我们有

$$\lim_{i \to +\infty} \left|\varphi^i(u_0)\right| = +\infty \tag{1}$$

对于 u $= u_0u_1u_2\ldots$ with φ(u) = u。这样的词 u 是该态射φ的一个 不动点 。这一概念被形式化为一个局部语境。

local e 不动_点 = 端同态 +**固定** u**假设**不动_点:
"(morph_streamf u) = u" **且**shd_grow: "
growing f [shd u]"

第二个假设恰好对应于条件（1）。在这个局部语境中，我们确实可以证明，不动点 $\mathcal{F}$ u 的语言等于态射 f 的纯形态语言，其中公理等于由 1 的首字母组成的长度为 1 的词 u:

引理 wlan_pmor: "$\mathcal{F}$ u = 纯_形态_语言 f ([shd u])"

纯周期无限词，即形如 $uuu\ldots$ 的词，实际上已作为 cycle u 存在于 “HOL-Library.Stream” 中，其中 u 是循环节。我们以一种直接的方式定义这一概念:

定义 纯周期_周期**其中**
"纯周期_周期 uu $\equiv$ ∃u. uu = cycle u"

不难证明，纯周期词的因子复杂度最终为常数，且其上界为重复部分的共轭词数量。为此，我们使用组合学_词中的本原根映射。给定一个非空词 w，它返回满足 $w = (\rho(w))^k$ 的词 $\rho(w)$，其中 k取最大值。

引理 pperiodic_fac_comp:**假设** "w $\neq \varepsilon$"
表明 "因子_复杂度 (cycle w) n $\leq$ |ϱ w|" "n $\geq$ |ϱ w| $\Longrightarrow$ 因子_复杂度 (cycle w) n = |ϱ w|"

对于最终周期无限词，也给出了类似的定义和相应的命题。

5 斯特姆词

斯特姆词是无限词，在词组合学中是一个极受欢迎的研究主题。Morse 和 Hedlund 在 1940年 的文章中研究了它们[15], ，并且它们有许多等价刻画（参见 [2]）。在这种形式化中，我们选择一种构造性刻画，它可以对所有斯特姆词的集合进行参数化。更具体地说，我们选择具有无理斜率的下机械词概念。首先，需要为给定的实数参数 n 和 α 定义第 β 个元素:

定义 下_机械_词_n :: "实数 ⇒ 实数 ⇒ 整数 ⇒ 整数"**其中**

"较低_机械_词_n α β n ≡ ⌊(n+1)*α+β⌋ - ⌊n*α+β⌋"

随后，便使用上文提到的 snth_流无缝生成该无限词。

定义 较低_机械_词 :: "实数 ⇒ 实数 ⇒ 整数 流"**其中**

"较低_机械_词 α β ≡ snth_流 (较低_机械_词_n α β)"

实数并非 Isabelle 主软件包的一部分，因此必须导入理论 “HOL.Real”。它提供了足够的工具来处理实数，以形式化（并证明）以下断言：

引理 lmw_rat_per: **假设** "α ∈ ℚ "**得出** "纯周期的（下机械词）"

引理 lmw_per_rat: **假设** "（下机械词__ $\alpha\beta$）是周期的"

得出 "α ∈ ℚ"|

这些命题展示了一个众所周知的事实：如果 α 是有理数，那么斜率为 α 的下机械词是周期的，反之亦然。因此，如果 α 是无理数，词 lower_mechanical_word α β 就是一个非周期词，也就是一个斯特姆词。

最后两个引理的证明借助单位圆的旋转完成；事实上，这种方法也用于斯特姆词的另一种已知刻画。

6 形式化的结构

前3节中介绍的形式化内容可在“形式化词组合学”项目的代码库中找到[7]，作为“CoW_Infinite”软件包的一部分，分为3个理论：

1. `CoW_Infinite/Languages.thy`介绍一般语言和（纯）形态语言，详见第3节；
2. `CoW_Infinite/Infinite_Words.thy`介绍无限词，详见第4节；3. `CoW_Infinite/Sturmian_Words.thy`介绍下机械词，详见第5节。

前两个理论旨在作为“形式化的词组合学”项目扩展至无限词领域的一般框架，而最后一个关于斯特姆词的理论则是一个概念验证，用以展示这些框架会话在一个热门主题上运行良好。

致谢

作者感谢捷克科学基金会提供的资助，项目编号为 GAČR 20-20621S。作者还向“形式化的词组合学”项目的其他作者 Štěpán Holub 和 Martin Raška 表达了谢意。

参考文献

1. 形式化证明档案。https://www.isa-afp.org/topics.html 2. Balková, L., Pelantová, E., Starosta, Š.: 多字母表上的斯图姆丛林（还是花园？）。RAIRO-Theor. Inf. Appl. 44, 443–470（2010）。https://doi.org/10 2011002 3 1051. /ita/ . Berstel, J., Perrin, D.: 词组合学的起源。《欧洲组合学杂志》28, 996 1022 2007 10 1016 2005 07 019 – ( )。https://doi.org/http://dx.doi.org/ . /j.ejc. . . 4. Ehrenfeucht, A., Rozenberg, G.: DOL 语言中子词的重复。Information and Control 59(1), 13–35（1983）。https://doi.org/https://doi.org/10.1016/S0019-9958(83)80028-X, https://www.sciencedirect.com/science/article/pii/S001999588380028X 5. Holub, Š., Raška, M., Starosta, Š.: 形式化的词组合学。https://gitlab.com/formalcow/combinatorics-on-words-formalized 6. Holub, Š., Raška, M., Starosta, Š.: 不保持本原性的二进制码。载于：Blanchette, J., Kovács, L., Pattinson, D.（编）《自动推理》。第369–387页。施普林格国际出版公司，尚（2022）7. Holub, Š., Raška, M., Starosta, Š.: 形式化的词组合学 v1.8。https://gitlab.com/formalcow/combinatorics-on-words-formalized（2023）8. Holub, Š., Starosta, Š.: 词组合学基础的形式化。载于：Cohen, L., Kaliszyk, C.（编）第12届交互式定理证明国际会议（ITP 2021）。莱布尼茨国际信息学会议录（LIPIcs），第193卷，第22:1–22:17页。Schloss Dagstuhl – 莱布尼茨信息学中心，德国达格施图尔（2021）。https://doi.org/10.4230/LIPIcs.ITP.2021.22, https://drops.dagstuhl.de/opus/volltexte/2021/13917 9. Holub, Š., Raška, M., Starosta, Š.: 词组合学基础。形式化证明档案（2021年5月），https://isa-afp.org/entries/Combinatorics_Words.html，形式化证明开发 10. Holub, Š., Starosta, Š.: 在 Isabelle/HOL 中形式化林登词。载于：语言理论进展：第25届国际会议，DLT 2021，葡萄牙波尔图，2021年8月16–20日，会议论文集。第217—228页。Springer-Verlag，柏林，海德堡（2021 10 1007 978 3 030 81508 0_18），https://doi.org/ . / - - - - . Isabelle 通用证明助理。https://isabelle.in.tum.de/ 12. Kaliszyk, C., Rabe, F.: 数学形式化语言综述。载于：Benzmüller, C., Miller, B.（编）《智能计算机数学》。第138–156页。施普林格国际出版公司，尚（2020）13. Lindenmayer, A.: 细胞发育相互作用的数学模型 I。具有单侧输入的丝状体。《理论生物学杂志》18(3), 280–299（1968）。https://doi.org/https://doi.org/10.1016/0022-5193(68)90079-9, https://www.sciencedirect.com/science/article/pii/0022519368900799

14. Lothaire, M.:《词组合学》，数学及其应用百科全书，第17卷。Addison-Wesley，马萨诸塞州雷丁（1983），后于剑桥数学丛书中重印，剑桥大学出版社，英国剑桥，1997年 15. Morse, M., Hedlund, G.A.: 符号动力系统 II——Sturmian轨道。《美国数学杂志》**62**，1–42（1940）16. Nikpow, T.: 结构化 Isar 证明教程简介。https://courses.grainger.illinois.edu/cs576/sp2015/doc/isar-overview.pdf 17. Nipkow, T., Wenzel, M., Paulson, L.C.（编）：Isabelle/HOL。施普林格，柏林，海德堡（2002年）18. Prouhet, E.: 关于数的幂之间若干关系的回忆录。《巴黎科学院院报》**33**，225（1851年）19. Rozenberg, G., Salomaa, A.:《L之书》。施普林格（1986年）。https://doi.org/103 201007978364295486./----. Thue, A.: 关于无限字符序列。数学与自然科学班论文集（1906年）21. Wiedijk, F.: 形式化100个定理。https://www.cs.ru.nl/~freek/100/