arXiv:2301.10198v1 [cs.FL] 24 Jan 2023

# 具有反转有界计数器的机器的无界性问题

Pascal Baumann[1][0000−0002−9371−0807]、Flavio D'Alessandro[2],Moses Ganardi[1][0000−0002−0775−7781]、Oscar Ibarra[3],Ian McQuillan[4],Lia Schütze[1][0000−0003−4002−5491]，以及 Georg Zetzsche[1][0000−0002−6421−4388]

[1] 马克斯·普朗克软件系统研究所（MPI-SWS），德国 [2] 罗马第一大学 G. Castelnuovo 数学系，意大利 [3] 加利福尼亚大学圣巴巴拉分校计算机科学系，美国加利福尼亚州圣巴巴拉 [4] 萨斯喀彻温大学计算机科学系，加拿大萨斯卡通

**摘要**。 我们研究一类关于形式语言的一般判定问题，称为"（一维）无界性谓词"，其对象是具有反转有界计数器的自动机（RBCA）。我们证明，这一类中的每个问题都可在多项式时间内以非确定性方式归约到有限自动机的同一问题。我们还证明，对于既能使用下推栈又能使用反转有界计数器的自动机（PRBCA），也存在类似的归约。

这使我们能够回答若干未决问题：例如，我们证明，判定给定的（P）RBCA 语言 $L$ $w_1 \ldots w_n$ $L \subseteq$ 是否有 $w_1^* \cdots w_n^*$ 界是 coNP 完全问题，也就是说，是否存在字符串 ,,使得 。对于 PRBCA，甚至其可判定性此前都是未决的。我们的方法还表明，不存在具有中间增长的（P）RBCA 语言。这意味着，每种长度的字符串数量要么呈多项式增长，要么呈指数级增长。我们证明中的一部分可能具有独立的研究价值：我们证明，可以在对数空间内将 RBCA 转换为具有 $\mathbb{Z}$ 个计数器的机器，同时保持其接受的语言不变。

**关键词：** 形式语言 · 可判定性 · 复杂性 · 计数器自动机 · 反转有界 · 下推 · 有界性 · 无界性

## 1 引言

形式语言理论中的一个经典思想是语言有界性的概念。字母表 $\Sigma$ 上的语言 $L$ 若满足以下条件，则称为有界： 存在一个数 $n \in \mathbb{N}$ 和一些词 $w_1, \ldots\ldots, w_n \in \Sigma^*$，使得 $L \subseteq w_1^* \cdots w_n^*$。有界性之所以重要，是因为对于有界语言，许多种算法问题都变得可判定。例如，Ginsburg 和 Spanier [26] 于 1964年引入有界性时，就已证明：给定两个上下文无关语言，其中一个有界，可以判定包含关系 [26,定理 6.3]。这是因为，如果 $L \subseteq w_1^* \cdots w_n^*$ 对于一个上下文无关语言成立，那么集合 $\{(x_1, \ldots\ldots, x_n) \in \mathbb{N}^n \mid w_1^{x_1} \cdots w_n^{x_n} \in L\}$ 是有效半线性的，因此可以用 Presburger 算术来表达包含关系。这里，有界性是一个

关键假设：Hopcroft 已证明，若 $L_0 \subseteq \Sigma^*$ 是上下文无关的，那么对于给定的上下文无关语言 $L$，判定 $L_0 \subseteq L$ 的问题可判定，当且仅当 $L_0$ 有界 [36, 定理 3.3]。

将关于有界语言的问题转换为 Presburger 算术问题这一思路，已应用于其他若干领域。例如，Esparza、Ganty 和 Majumdar [20] 已经证明，许多类无限状态系统都在有界语言模意义下是完备的，这意味着有界语言构成了一个子类，适用于许多算法问题。再如，在有界上下文无关语言上，子词序的一阶理论是可判定的 [46], ，而在语言上 $\Sigma^*$，甚至存在理论都是不可判定的 [34]。这进而意味着，对于有界上下文无关语言上的子词序，初始极限 Datalog 是可判定的 [6]。最后，有界上下文无关语言可以用正则语言进行紧密近似 [16]。

这引出了一个问题：如何判定给定语言是否有界？对于上下文无关语言，这个问题可以在多项式时间内判定 [26, 定理 5.2(a)] [25,定理 19]。

**RBCA 的有界性**。尽管有界性十分重要，但多年来，人们一直未能确定 [8, 17][5] 最受广泛研究的无限状态系统之一——反转有界（多）计数器自动机（RBCA）的有界性是否可判定。这类机器配有计数器，可以递增、递减，甚至可以测试计数器是否为零。然而，为了使基本问题可判定，每个计数器可反转的次数都受到一个界的限制，也就是说，计数器在递增和递减阶段之间切换的次数受到限制。它们最早于20世纪70年代得到研究 [2, 37] ，此后也受到了广泛关注 [7–11, 13, 18, 23, 29, 33, 34, 40–42, 59]。上述有界上下文无关语言的理想性质同样适用于有界 RBCA。此外，RBCA（即使配有栈）接受的任何有界语言都可以有效地确定化 [39]（另见 [8, 10]），为算法分析开辟了更多途径。因此，有界性的可判定性多年来一直悬而未决，实在令人意外。

RBCA 有界性的可判定性已在 [15], 中得到解决，该文证明，即使对于更大的带状态向量加法系统（VASS）类，采用按配置接受的方式，有界性仍然是可判定的。然而， [15]中的结果仍有若干方面尚未阐明，本文将对此进行研究：

Q1：判定 RBCA 的有界性有何复杂性？ [15] 中的算法采用了向量加法系统的 KLMST 分解 [44, 47, 49, 51, 55]„ 众所周知，这会导致阿克曼复杂度 [50]。Q2：下推反转有界计数器自动机（PRBCA）的有界性是否可判定 [37]? 这类自动机除了具有反转有界计数器外，还具有一个栈。它们可以对带有数值数据类型的递归程序进行建模 [33]。 [17, 18]曾指出，有界性是否可判定仍是一个开放问题。Q3：RBCA 是否存在具有中间增长的语言？据我们所知，这本身就是一个由来已久的开放问题 [38]。语言的增长

[5] 请注意， [8]讨论的是 Parikh 自动机，它们与 RBCA 等价。

$L \subseteq \Sigma^*$ 是计数函数 $g_L\colon\ \mathbb{N} \to \mathbb{N}$，其中 $g_L(n)$ 是 $L$中长度为 $n$ 的字符串数量。这个概念与有界性密切相关：对于正则语言和上下文无关语言，已知一个语言具有多项式增长，当且仅当它是有界的（否则它具有指数增长）。如果一个语言既不具有多项式增长，也不具有指数增长，则称其具有中间增长。

**贡献 I：**我们证明了该主要结果的两个版本：一个适用于 RBCA，另一个适用于 PRBCA。具体而言，论文 [15] 不仅证明了向量加法系统的有界性是可判定的，还为形式语言引入了一类通用的无界性谓词。随后， [15] 证明，对于向量加法系统，任意无界性谓词可判定，当且仅当它对于正则语言可判定。我们的前两个主要结果如下：

MR1：判定 RBCA 的任意无界性谓词，可在 NP 时间内归约为判定正则语言的同一谓词。MR2：判定 PRBCA 的任意无界性谓词，可在 NP 时间内归约为判定上下文无关语言的同一谓词。

不过，需要指出的是，我们的结果仅适用于 [15] 中的一维无界性谓词。幸运的是，这些结果足以支持我们的应用。借助这些结果，我们得以解决上述问题 (Q1)–(Q3)，并确定其他若干问题的精确复杂性。由此可知，RBCA 和 PRBCA 的有界性都是 coNP 完全，从而回答了 (Q1) 和 (Q2)。此外，证明表明，如果 PRBCA 不满足有界性，那么其语言就具有指数增长。这意味着不存在具有中间增长的 RBCA 语言（从而解决了 (Q3)），而且 PRBCA 也同样如此。特别地，判定（P）RBCA 的多项式增长是 coNP 完全的，而判定（P）RBCA 的指数增长是 NP 完全的。我们还可以从结果推出，判定（P）RBCA 语言是否无限是 NP 完全的（不过，这也可由 [33], 参见第 2 节轻易得出）。最后，我们的结果表明，判定一个 RBCA 语言是否因子普适是 PSPACE 完全的，也就是说，该语言包含 $\Sigma^*$ 的每个词作为因子（即作为中缀）。RBCA 的这一问题是否可判定，也曾在 [17, 18] 中作为一个未决问题提出（称为中缀密度）。

我们首先将（P）RBCA 转换为使用 $\mathbb{Z}$ 计数器而非反转有界计数器的模型，以此证明我们的结果（MR1）和（MR2）。 $\mathbb{Z}$-计数器可以递增和递减，但不能进行零测试。此外，它可以取负值。对于这类计数器，若能到达所有计数器均为零的配置，则接受（特别是，接受条件允许对每个计数器进行一次零测试）。在这里，带有 $\mathbb{Z}$计数器的有限自动机称为 $\mathbb{Z}$-$VASS$ [30]。 $\mathbb{Z}$计数器也称为盲计数器 [27] ，而且有一个标准结论：RBCA 与 $\mathbb{Z}$-VASS [27,等价（就接受的语言而言）定理 2]。

尽管 RBCA 与 $\mathbb{Z}$-向量加法系统之间的等价性早已广为人知，但显然此前并没有已知的、能在多项式

| 问题 | $\mathbb{Z}$-向量加法系统/RBCA | $\mathbb{Z}$-文法/PRBCA |
|---|---|---|
| 有界性 | coNP 完全 | coNP 完全 |
| 有限性 | coNP 完全 | coNP 完全 |
| 因子普遍性 | PSPACE 完全，不可判定 | |

**表 1.** 复杂度结果。完备性结论均指相对于确定性对数空间归约而言。

时间内将 RBCA 翻译为 $\mathbb{Z}$-向量加法系统的方法。这里的难点在于模拟零测试（在 RBCA 中，零测试可能发生无界多次）：为了模拟这些零测试，$\mathbb{Z}$-向量加法系统需要仅用多项式级数量的控制状态，跟踪每个计数器分别完成了哪些递增／递减阶段。此外，如何利用 $\mathbb{Z}$-计数器来做到这一点也并不明显，因为这些计数器只会在最后进行检查。

**贡献 II：** 作为证明 (MR1) 的第一步，我们证明

MR3：RBCA 可以在对数空间内翻译为 $\mathbb{Z}$-向量加法系统（同时保持语言不变）。

这也意味着，可以在多项式时间内翻译为另一种等价模型 Parikh 自动机 [42], ，也可以从该模型翻译而来：近期研究表明，Parikh 自动机（近年来受到广泛关注 [5, 7–9, 13, 22]）可以在多项式时间内翻译为 $\mathbb{Z}$-向量加法系统 [31]。结合我们的新结果可知，可以在 RBCA、$\mathbb{Z}$-向量加法系统和 Parikh 自动机之间进行多项式时间翻译。此外，我们的结果还给出了将 PRBCA 翻译为 $\mathbb{Z}$-文法的对数空间算法；这种文法是带有 $\mathbb{Z}$-计数器的上下文无关文法的扩展。后者是证明 (MR2) 的第一步。

## 2 主要结果：无界性与（P）RBCA

**反转有界计数器自动机与下推自动机。** 带有 $k$ 计数器的下推自动机是一个元组 $\mathcal{A} = (Q, \Sigma, \Gamma, q_0, T, F)$，其中 $Q$ 是状态的有限集，$\Sigma$ 是输入字母表，$\Gamma$ 是栈字母表，$q_0 \in Q$ 是初始状态，$T$是转移的有限集（$p, w, \mathsf{op}\ q$），$\in Q \times \Sigma^* \times\ \mathsf{Op}\ \times Q$，且 $F \subseteq Q$是终态集。这里，Op 定义为

$$\mathrm{Op} = \{\mathsf{inc}_i, \mathsf{dec}_i, \mathsf{zero}_i, \mathsf{nz}_i \mid 1 \le i \le k\} \cup \Gamma \cup \bar{\Gamma} \cup \{\varepsilon\},$$

¯包含计数器和栈操作。这里 $\Gamma = \{\bar{\gamma} \mid \gamma \in \Gamma\}$ 是 $\Gamma$的一个不相交副本。一个配置是一个元组 $(p, \alpha, \boldsymbol{v}) \in Q \times \Gamma^* \times \mathbb{N}^k$。如果存在一个 $(p, \alpha, \mathsf{op}\,\boldsymbol{u}) \xrightarrow{w}$，使得以下情况之一成立，我们就写作 $(p', \alpha', \boldsymbol{u}')$：

- $\mathrm{op} = \mathsf{inc}_i$, $\boldsymbol{u}' = \boldsymbol{u} + \boldsymbol{e}_i$, and $\alpha' = \alpha$ where $\boldsymbol{e}_i \in \mathbb{N}^k$ is the $i$-th unit vector,
- $\mathrm{op} = \mathsf{dec}_i$, $\boldsymbol{u}' = \boldsymbol{u} - \boldsymbol{e}_i$, and $\alpha' = \alpha$,
- $\mathrm{op} = \mathsf{zero}_i$, $\boldsymbol{u}[i] = 0$, $\boldsymbol{u}' = \boldsymbol{u}$, and $\alpha' = \alpha$

- $\mathrm{op} = \mathsf{nz}_i$, $\boldsymbol{u}[i] \neq 0$, $\boldsymbol{u}' = \boldsymbol{u}$, and $\alpha' = \alpha$,
- $\mathrm{op} = \gamma \in \Gamma$, $\boldsymbol{u}' = \boldsymbol{u}$, and $\alpha' = \alpha\gamma$,
- $\mathrm{op} = \bar{\gamma} \in \bar{\Gamma}$, $\boldsymbol{u}' = \boldsymbol{u}$, and $\alpha'\gamma = \alpha$,
- $\mathrm{op} = \varepsilon$, $\boldsymbol{u}' = \boldsymbol{u}$, and $\alpha' = \alpha$.

我们以自然的方式将这一记法扩展到更长的运行。

一个 $(k, r)$-$PRBCA$ （下推反转有界计数器自动机）$(\mathcal{A}, r)$由一个下推自动机、 $k$ 个计数器 $\mathcal{A}$以及一个以一元编码的数 $r \in \mathbb{N}$组成。如果上一次影响计数器 $\mathsf{c}_i$ （非测试）操作是 $\mathsf{inc}_i$ ，而下一次操作是 $\mathsf{dec}_i$，或反之，则该计数器会反转。如果每个计数器反转的次数都不超过 $r$ 次，则称一次运行是-反转有界的。$(\mathcal{A}, r)$的语言是

$$L(\mathcal{A}, r) = \{w \in \Sigma^* \mid \exists q \in F,\ r\text{-反转有界运行 } (q_0, \varepsilon, \mathbf{0}) \xrightarrow{w} (q, \varepsilon, \mathbf{0})\}.$$

一个 $(k, r)$-$RBCA$ （反转有界计数器自动机）是一个 $(k, r)$-PRBCA，其中 $\mathcal{A}$ 仅使用计数器操作。我们用 RBCA 和 PRBCA 表示（P）RBCA 语言类。

请注意，我们在外部 施加反转界（遵循 [33]），而文献中的其他定义则要求自动机在内部 确保每次（接受）运行中的反转次数不超过 $r$，例如 [37]。显然，我们的定义涵盖了后一种定义；特别是，定理1对具有内部检查反转界的（P）RBCAs也成立。

一个 $d$ 维 $\mathbb{Z}$-向量加法系统（$\mathbb{Z}$，带状态向量加法系统）是一个元组$\mathcal{V} = (Q, \Sigma, q_0, T, F)$，其中 $Q$ 是有限状态集， $\Sigma$ 是字母表， $q_0 \in Q$ 是初始状态， $T$ 是有限转移集 $(p, w, \boldsymbol{v}, p') \in Q \times \Sigma^* \times \mathbb{Z}^d \times Q$，而 $F \subseteq Q$ 是终态集。 $\mathbb{Z}$-向量加法系统的配置是一个元组 $(p, \boldsymbol{v}) \in Q \times \mathbb{Z}^d$。如果存在转移 $(p, w, \boldsymbol{v}, p')$ 使得 $\boldsymbol{u}' = \boldsymbol{u} + \boldsymbol{v}$，我们就记作 $(p, \boldsymbol{u}) \xrightarrow{w} (p', \boldsymbol{u}')$。我们以自然的方式将此记号扩展到更长的运行。 $\mathbb{Z}$-向量加法系统的语言定义为

$$L(\mathcal{V}) = \{w \in \Sigma^* \mid \exists q \in F\colon\ (q_0, \mathbf{0}) \xrightarrow{w} (q, \mathbf{0})\}.$$

一个（$d$维） $\mathbb{Z}$-文法是一个元组 $G = (N, \Sigma, S, P)$，其中 $N$ 和 $\Sigma$ 是互不相交的有限集，分别包含非终结符和终结符； $S \in N$是开始非终结符； $P$是一个有限产生式集，其产生式形式为 $(A, u, \boldsymbol{v}) \in N \times (N \cup \Sigma)^* \times \mathbb{Z}^d$。我们也用 $(A \to u, \boldsymbol{v})$ 代替 $(A, u, \boldsymbol{v})$。我们称 $\boldsymbol{v}$ 为产生式 $(A \to u, \boldsymbol{v})$ 的（计数器）效应。对于词 $x, y \in (N \cup \Sigma)^*$，若存在产生式 $(A \to u, \boldsymbol{v})$，使得 $x = rAs$ 且 $y = rus$，则记作 $x \Rightarrow^{\boldsymbol{v}} y$。此外，若存在词 $x_1, \ldots, x_n \in (N \cup \Sigma)^*$ 以及 $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n \in \mathbb{Z}^d$，满足 $x \Rightarrow^{\boldsymbol{v}_1} x_1 \Rightarrow^{\boldsymbol{v}_2} \cdots \Rightarrow^{\boldsymbol{v}_n} x_n = y$且 $\boldsymbol{v} = \boldsymbol{v}_1 + \cdots + \boldsymbol{v}_n$，则记作$x \overset{*}{\Rightarrow}^{\boldsymbol{v}} y$。如果计数器效应无关紧要，我们使用记号 $\Rightarrow$：若存在 $\boldsymbol{v}$ 使得$x \Rightarrow_{\boldsymbol{v}} y$，则有 $x \Rightarrow y$；对于 $\overset{*}{\Rightarrow}$，类似可得。若推导仅限于产生式的子集 $Q \subseteq P$，则记作 $\Rightarrow_Q$ （分别为 $\overset{*}{\Rightarrow}_Q$）。

$\mathbb{Z}$-文法 $G$ 的语言 是满足$S \overset{*}{\Rightarrow}^{\mathbf{0}} w$的所有字符串 $w \in \Sigma^*$ 的集合。换句话说，如果存在一个推导 $S \overset{*}{\Rightarrow} w$，使得所有出现的产生式的效应之和为零向量 $\mathbf{0}$，则该推导成立。维度为$d$ 的 $\mathbb{Z}$-文法也称为上的价文法 $\mathbb{Z}^d$ [21]。

就我们的目的而言，只需假设 $\mathbb{Z}$-向量加法系统和 $\mathbb{Z}$-文法中出现的 $\mathbb{Z}^d$-维向量（效应）采用一元编码即可。不过，这并不构成限制：使用一元编码可以轻松模拟采用 $n$ 位二进制编码的数进行的计数器更新，代价是需要 $dn$ 多个新计数器（见附录 A.1）。

**转换结果**。 下面给出我们的第一个主要定理：

**定理1**。 *RBCA* 可在对数空间内转换为 $\mathbb{Z}$-向量加法系统。*PRBCA* 可在对数空间内转换为 $\mathbb{Z}$-文法。

这里的转换指保持所接受（或生成）的语言不变的翻译。有几种机器模型在所接受语言方面与 RBCA 等价。借助定理1，我们给出了最后一个尚缺的翻译：

**推论1**。 以下模型可以在对数空间内相互转换：*(i) RBCA*，*(ii)* $\mathbb{Z}$-向量加法系统，*(iii)* 采用 $\exists$*PA* 接受条件的 *Parikh* 自动机，以及 *(iv)* 采用半线性接受条件的 *Parikh* 自动机。

粗略地说，Parikh 自动机是一种只能递增计数器的机器。如果一次运行最终的计数器值属于某个半线性集，则该运行就是接受的。Klaedtke 和 Rueß 引入了 Parikh 自动机 [42]，其中接受条件用半线性表示法（由基向量和周期向量组成）指定，从而得到上述模型 (iv)。例如， [34], 也可以用存在型 Presburger 公式（简称 ∃PA）指定接受条件，从而得到上述模型 (iii)。定理1证明了 (i)⇒(ii)，而 (ii)⇒(i) 则显而易见（ [41, 定理4.5]给出了一种巧妙且高效的翻译）。此外，(ii)⇒(iii) 和 (ii)⇒(iv) 也很直接。对于 (iii)⇒(ii)，可以按照 [31, 命题 V.1]，中的方法进行，而 (iv)⇒(ii) 也很简单。

**无界性谓词**。 我们将利用定理1证明第二个主要定理，该定理涉及 [15] 中引入的无界性谓词。在 [15], 中，无界性谓词可以是一维的，也可以是多维的，但在本文中，我们只考虑一维无界性谓词。

设 $\Sigma$ 是一个字母表。（语言）谓词 是 $\Sigma$ 上语言的一个集合。若$\mathfrak{p}$ 是一个谓词，且 $L \subseteq \Sigma^*$ 是一个语言，则我们用 $\mathfrak{p}(L)$ 表示 $\mathfrak{p}$ 对语言 $L$ 成立（即 $L \in \mathfrak{p}$）。若谓词 $\mathfrak{p}$ 满足以下条件（对所有 $K$，$L \subseteq \Sigma^*$），则称其为一个（一维）无界性谓词：

(U1) 若 $\mathfrak{p}(K)$ 且 $K \subseteq L$，则 $\mathfrak{p}(L)$。(U3) 若 $\mathfrak{p}(K \cdot L)$，则 $\mathfrak{p}(K)$ 或 $\mathfrak{p}(L)$。(U2) 若 $\mathfrak{p}(K \cup L)$，则 $\mathfrak{p}(K)$ 或 $\mathfrak{p}(L)$。(U4) 当且仅当 $\mathfrak{p}(F(L))$ 时，$\mathfrak{p}(L)$。

这里 $F(L) = \{v \in \Sigma^* \mid \exists u, w \in \Sigma^* \colon uvw \in L\}$ 是 $L$的因子集合（有时也称为内因子）。特别地，最后一个条件表明，$\mathfrak{p}$ 只取决于某种语言中出现的因子集合。

对于无界性谓词 $\mathfrak{p}$ 和有限表示语言类 $\mathcal{C}$ （例如自动机或文法），令 $\mathfrak{p}(\mathcal{C})$ 表示这样一个问题：判定来自 $\mathcal{C}$ 的给定语言 $L$ 是否满足 $\mathfrak{p}$ 。形式化地说，$\mathfrak{p}(\mathcal{C})$ 是以下判定问题：

**给定**一个语言 $L$，来自 $\mathcal{C}$。
**问题**Does $\mathfrak{p}(L)$ hold?

例如，$\mathfrak{p}$(RBCA) 是判定反转有界多计数器自动机是否满足 $\mathfrak{p}$ 的问题，而 $\mathfrak{p}$(NFA) 是判定非确定有限自动机是否满足 $\mathfrak{p}$ 的问题。我们指出，公理 (U1)–(U4) 略强于 [15], 中使用的公理，但两种定义所产生的判定问题集合相同（因为在 [15], 中，人们总是判定 $\mathfrak{p}(F(L))$ 是否成立）。因此，定理 2 的表述不受所用定义的影响。详情见附录 A.2。

以下是语言 $L \subseteq \Sigma^*$ 的（一维）无界性谓词示例，这些谓词已在 [15] 中得到确立。我们在此列出它们，以便直观了解我们的结果适用范围：

**非有界**当且仅当 $L$ 是非有界语言时，令 $\mathfrak{p}_{\text{notb}}(L)$。**非空性**当且仅当 $L \neq \emptyset$ 时，令 $\mathfrak{p}_{\neq\emptyset}(L)$。**无限性**当且仅当 $L$ 是无限的时，令 $\mathfrak{p}_{\infty}(L)$。**因子全体性**当且仅当 $\Sigma^* \subseteq F(L)$ 时，令 $\mathfrak{p}_{\text{funi}}(L)$。

不难证明这些都是无界性谓词，但证明可见于 [15]。以下是我们的第二个主要定理：

**定理 2**。设 $\mathfrak{p}$ 是一维无界性谓词。存在一个从 $\mathfrak{p}$(PRBCA) 到 $\mathfrak{p}$(下推自动机) 的NP 归约。此外，存在一个从 $\mathfrak{p}$(RBCA) 到 $\mathfrak{p}$(非确定有限自动机) 的NP 归约。

这里，从问题 $A \subseteq \Sigma^*$ 到 $B \subseteq \Sigma^*$ 的 NP 归约 是一个非确定性多项式时间图灵机，使得对于每个输入词 $w \in \Sigma^*$，$w \in A$ 成立，当且仅当存在该图灵机的一次运行，产生一个属于 $B$ 的词。

现在来看定理 2 的一些应用，另请参见表 1。以下完备性结果均是针对确定性对数空间归约而言的。

**推论 2**。 PRBCA 和 RBCA 的有界性是coNP完全的。

对于推论 2，我们证明判定非有界性 是 NP 完全的。为此，我们将定理 2 应用于谓词 $\mathfrak{p}_{\text{notb}}$，从而得到一个 NP 上界，因为上下文无关语言的有界性可以在多项式时间内判定 [25]。NP 下界直接来自 RBCA 非空性问题的 NP 困难性 [29, 定理 3]，因此 PRBCA 也一样。

**推论 3.** PRBCA 和RBCA 的coNP完全性。

我们通过证明检验无限性 是 NP 完全的，来证明推论 3。上界可通过谓词 $\mathfrak{p}_{\infty}$ 由定理 2 得出。与前文一样，NP 困难性源自 RBCA 和 PRBCA 的非空性问题。

不过，推论 3 中的结果并不新颖。它们可直接由以下事实推出：对于给定的 PRBCA（或 RBCA），可以在多项式时间内构造一个存在型 Presburger 算术（∃PA）公式来表示其 Parikh 像；RBCA 的相关结果见 [37]，PRBCA 的相关结果见 [33]。关于 ∃PA，有一项标准结果指出，对于每个公式 $\varphi$，都存在一个界 $B$，使得 (i) $B$ 至多为

关于 $\varphi$ ，呈指数级增长；并且 (ii) $\varphi$ 定义一个无限集，当且仅当存在某个向量，其某个分量大于 $B$，使得 $\varphi$成立。例如，这可以从 [54]推导出来。因此，可以很容易地构造第二个 ∃PA 公式 $\varphi'$，使得 $\varphi$ 定义一个无限集，当且仅当 $\varphi'$ 可满足。

**推论 4.** *RBCA* 的因子普遍性是 PSPACE 完全问题。

RBCA 的因子普遍性是否可判定，在 [17, 18] 中仍被列为一个开放问题（在那里使用的术语是中缀密度）。推论 4 可利用 $\mathfrak{p}_{\mathsf{funi}}$ 由定理 2 得出，因为非确定有限自动机的因子普遍性是 PSPACE 完全问题：要判定对于正则语言 $R$， $\Sigma^* \subseteq F(R)$ 是否成立，我们只需计算一个自动机来识别 $F(R)$，并在 PSPACE 内检查包含关系。对于下界，可以归约非确定有限自动机的 PSPACE 完全普遍性问题，因为对于 $R \subseteq \Sigma^*$，语言 $(R\#)^* \subseteq (\Sigma \cup \{\#\})^*$ 是因子普适的，当且仅当 $R = \Sigma^*$。请注意，单计数器语言 [18], 的因子普遍性已知是不可判定的，因此 PRBCA 的因子普遍性也不可判定。然而，对于栈反转次数有界的下推自动机 [18]，该问题是可判定的。

**超越下推自动机**。 定理 2 引出了这样一个问题：对于任意一类$\mathcal{M}$ 机器，是否都能将扩展了反转有界计数器的 $\mathcal{M}$ 机器的任意无界性谓词，归约为仅针对 $\mathcal{M}$机器的同一谓词。答案是否定的。例如，考虑二阶下推自动机，简称二阶下推自动机（2-PDA）。若通过添加反转有界计数器对其进行扩展，就得到二阶部分逆转有界计数器自动机（2-PRBCA）。对于二阶下推自动机（2-PDA），无限性问题是可判定的 [35] （更强的结果见 [3, 4, 14, 32, 53, 57]）。然而，二阶部分逆转有界计数器自动机（2-PRBCA）这一类甚至不具有可判定的空性问题，更不用说可判定的无限性问题了。 [58, 命题 7]证明了这一点（另一种证明见 [43, 定理 4] ）。因此，二阶部分逆转有界计数器自动机（2-PRBCA）的无限性问题无法归约为二阶下推自动机（2-PDA）的无限性问题。

**增长**。 最后，我们采用定理 2 证明中的方法，展示 RBCA 所接受语言的增长行为具有二分性。对于字母表 $\Sigma$，我们用 $\Sigma^{\leq m}$ 表示长度至多为 $m$ 的所有 $\Sigma$ 上的词的集合。如果存在一个多项式$p(x)$，使得对于所有 $m \geq 0$， $|L \cap \Sigma^{\leq m}| \leq p(m)$，那么我们称语言 $L \subseteq \Sigma^*$ 具有多项式增长[6] 。多项式增长的语言也称为稀疏 或多项式细长。如果存在一个实数 $r > 1$ ，使得对于无穷多个 $m$， $|L \cap \Sigma^{\leq m}| \geq r^m$ ，那么我们称 $L$ 具有指数增长 。显然，形如 $w_1^* \cdots w_n^*$ 的语言具有多项式增长，因此众所周知，有界语言具有多项式增长。我们证明：(a) 在 PRBCA 语言（尤其是 RBCA 语言）中，反过来同样成立；(b) 所有其他语言都具有指数增长（与某些模型不同，例如二阶下推自动机 [28], ，在这些模型中这种二分性并不成立）：

**定理 3.** 设 $L$ 是一个由 *PRBCA* 接受的语言。当且仅当 $L$ 有界时， $L$ 具有多项式增长。如果 $L$ 无界，则其呈指数增长。

[6] 在 [25], 中，多项式增长和指数增长的定义是用 $\Sigma^m$ 代替 $\Sigma^{\leq m}$，但这会导出等价的概念，参见附录 D

## 3 将反转有界计数器转换为 $\mathbb{Z}$-计数器

**将反转次数减少到一次**。本节证明定理 1，即从 RBCA 到 $\mathbb{Z}$-向量加法系统的转换。在 [29,引理 1],中声称，给定一个 $(k, r)$-RBCA，可以在关于 $k$ 和 $r$ 的多项式时间内构造一个接受相同语言的 $(k\lceil(r+1)/2\rceil,1)$-RBCA。他们提供的参考文献 [2] 确实包含了这样的构造 [2, 定理 5]的证明。[2] 中的构造只是一个粗略的概述，并未对复杂性作出任何说明；但据我们对该构造的理解，它会在状态中记录每个计数器的反转次数，这将导致指数级膨胀。

相反，我们按如下方式进行。考虑一个具有计数器 $\mathsf{c}_1, \ldots, \mathsf{c}_k$ 的 $(k, r)$-RBCA。不失一般性，假设 $r = 2m - 1$。我们将构造一个等价的 $(2k(r+1),1)$-RBCA。观察单个计数器 $\mathsf{c}_i$ 的行为，我们可以将每个 $r$-反转有界运行分解为若干不含反转的子运行。我们称这些子运行为阶段，并按顺序编号，从 1 到至多 $2m$。编号为奇数（偶数）的阶段是正相位（负），在此期间 $\mathsf{c}_i$ 只递增（递减）。我们用 $m$ 个单次反转计数器 $\mathsf{c}_{i,1}, \ldots, \mathsf{c}_{i,m}$ 替换 $\mathsf{c}_i$，其中 $\mathsf{c}_{i,j}$记录计数器 $\mathsf{c}_i$ 在正相位 $2j-1$期间的递增次数。

然而，我们的机器需要跟踪各个计数器分别处于哪个阶段，以便知道当前要使用哪些计数器 $\mathsf{c}_{i,j}$。我们通过以下方式实现这一点：对于每个 $k$ 个计数器 $\mathsf{c}_i$，我们还额外设置一组 $2m = r+1$ 个“相位计数器” $\mathsf{p}_{i,1}, \ldots, \mathsf{p}_{i,2m}$，用于记录当前所处的阶段。这样总共有 $km + k(r+1) \leq 2k(r+1)$ 个计数器。我们通过将 $\mathsf{p}_{i,j}$ 设为 1，并对每个 $j' \neq j$ 将 $\mathsf{p}_{i,j'}$ 设为 0，来编码计数器 $\mathsf{c}_i$ 处于阶段 $j$。由于阶段只会递增，相位计数器同样是单次反转的。

使用非零测试时，自动机可以在任意时刻以非确定方式猜测并验证每个计数器当前所处的阶段。这样，它就能为每条指令选择正确的计数器 $\mathsf{c}_{i,j}$。当计数器 $\mathsf{c}_i$ 处于正相位$2j-1$时，对 $\mathsf{c}_i$ 的递增和递减模拟如下：

**递增** 递增$\mathsf{c}_{i,j}$
**递减** 进入下一个（负）阶段 $2j$；然后以非确定方式选取某个 $\ell \in [1, j]$，并将 $\mathsf{c}_{i,\ell}$ 递减。我们不能直接将 $\mathsf{c}_{i,j}$ 递减，因为我们可能已切换到阶段 $j$，而此时 $\mathsf{c}_i$ 的值非零，因此 $\mathsf{c}_i$ 可能会被递减到超过 $\mathsf{c}_{i,j}$ 所允许的程度。

当计数器 $\mathsf{c}_i$ 处于负阶段 $2j$ 时，我们按如下方式模拟递增和递减：

**递增** 进入下一个阶段 $2j+1$（除非 $j = m$；此时机器会阻塞），并将 $\mathsf{c}_{i,j+1}$ 递增。**递减** 以非确定方式选取某个 $\ell \in [1, j]$，并将 $\mathsf{c}_{i,\ell}$ 递减。

最后，要模拟对 $\mathsf{c}_i$ 的零测试，我们对所有计数器 $\mathsf{c}_{i,1}$，……，$\mathsf{c}_{i,m}$ 执行零测试；而要模拟对 $\mathsf{c}_i$ 的非零测试，我们以非确定方式从计数器 $\mathsf{c}_{i,1}$，……，$\mathsf{c}_{i,m}$ 中选取一个进行非零测试。

正确性可通过以下性质轻松验证。如果在某个时刻，$\mathsf{c}_i$处于阶段 $2j-1$ 或 $2j$，那么 (i) $\sum_{\ell=1}^{j} \mathsf{c}_{i,\ell} = \mathsf{c}_i$，(ii) 计数器 $\mathsf{c}_{i,1}$，…，$\mathsf{c}_{i,j}$

至多发生过一次反转，并且 (iii) 计数器 $\mathsf{c}_{i,\,j+1},\ \ldots,\ \mathsf{c}_{i,\,m}$ 尚未被触动（尤其是，它们的值为零）。此外，如果 $\mathsf{c}_i$ 处于正相位 $2j-1$，那么 $\mathsf{c}_{i,\,j}$ 尚未发生过任何反转。

请注意，该构造会将原系统的每个转移替换为 $\mathcal{O}(r)$ 个新转移（和状态）。因此，我们的构造只会使系统规模线性增长（若 $r$ 固定，则为常数级增长）。构造的详细内容见附录 B。

**从 1-反转过渡到 $\mathbb{Z}$-计数器。** 现在，我们将 $(k,1)$-RBCA 转换为 $\mathbb{Z}$-向量加法系统。1-反转有界计数器与 $\mathbb{Z}$-计数器的区别在于：(i) 非负计数器的值为 0 时若被递减，就会阻止该操作；(ii) 1-反转有界计数器允许进行（非）零测试。请注意，所有零测试都发生在首次递增之前或最后一次递减之后。所有非零测试都发生在首次递增与最后一次递减之间。

如果计数器的数量 $k$ 有界，则以下简单方案可行。$\mathbb{Z}$-向量加法系统存储哪些计数器尚未递增，以及哪些计数器今后不会再递增的信息。这些信息足以忠实地模拟计数器（就上述性质 (i) 和 (ii) 而言），并使状态空间扩大 $2^k \cdot 2^k$ 倍。后一类信息需要由自动机猜测，并通过验证所有计数器最终都为零来确认。

在一般情形下，我们引入一种 $\mathbb{Z}$-VASS 的变体，它可以在开始时猜测多项式数量的比特，并在整个运行过程中读取这些比特。带猜测功能的 $d$ 维 $\mathbb{Z}$-*VASS*（$\mathbb{Z}$-*VASSG*）的格式与 $d$ 维 $\mathbb{Z}$-VASS 几乎相同，不同之处在于每个转移还带有一个关于某个变量有限集 $X$ 的命题公式。若存在一个赋值 $\nu\colon X \to \{0,1\}$，以及某个 $q \in F$ 对应的接受运行 $(q_0,\,\mathbf{0}) \xrightarrow{w} (q,\,\mathbf{0})$，使得运行过程中出现的所有公式都由 $\nu$ 满足，则词 $w \in \Sigma^*$ 被 $\mathbb{Z}$-VASSG 接受。

我们必须消除 $(k,\,1)$-RBCA 的零测试和非零测试。零测试（或非零测试）是否成功，取决于计数器当前所处的阶段（以及最终每个计数器是否都为零；但我们假设接受条件能确保这一点）。每个计数器最多经历 4 个阶段：

1. 在首次递增之前，3. “递减阶段”，以及 2. “递增阶段”，4. 在最后一次递减之后。

因此，每次运行都可以分解为 $4k$ 个（可能为空的）段，在每个段中，没有计数器会改变其阶段。我们的思路是猜测每个段中每个计数器所处的阶段。因此，我们有命题变量 $\mathsf{p}_{i,j,\ell}$，其中 $i \in [1,4k]$、$j \in [1,k]$ 和 $\ell \in [1,4]$。然后，当且仅当在段 $i$ 中，计数器 $j$ 处于阶段 $\ell$ 时，$\mathsf{p}_{i,j,\ell}$ 为真。我们必须检查该赋值对于每个计数器都是可接受的，也就是说，每个计数器的阶段序列都遵循上述顺序。

我们按如下方式修改机器。在其状态中，机器保留一个数值 $i \in [1,4k]$，用于记录当前段。运行开始时，机器使用一个命题公式检查赋值 $\nu$ 是否可接受：它检查

(i) 对于每个段 $i$ 和每个计数器 $j$，恰好存在一个阶段 $\ell$，使得 $\mathrm{p}_{i,\,j,\,\ell}$ 为真；并且 (ii) 遵循上述阶段顺序。随后，对于每个计数器操作，机器都会检查该操作是否与当前段一致。此外，如果当前操作需要切换段，则段计数器 $i$ 递增。例如，如果某个计数器在阶段 1 被递增，它就切换到阶段 2，段计数器也随之递增；或者，如果某个计数器在阶段 3 被测试是否为零，它就切换到阶段 4，段计数器也随之递增。

通过这些修改，我们可以检查与当前片段对应的变量来进行零测试：零测试只能在阶段 1 和 4 成功。类似地，对于非零测试，我们可以检查计数器是否处于阶段 2 或 3。

**将 $\mathbb{Z}$-向量加法系统图转换为 $\mathbb{Z}$-向量加法系统。** 为了处理上文提到的一般情形，我们需要说明如何将 $\mathbb{Z}$-向量加法系统图转换为普通的 $\mathbb{Z}$-向量加法系统。作为准备步骤，我们确保每个公式都是一个文字。将标记为公式 $\varphi$ 的转移替换为一个串并联图：先通过将否定向内推，把 $\varphi$ 转换为否定范式，然后可以用串联组合替换合取，用并联组合（非确定性）替换析取。

$\mathbb{Z}$-向量加法系统的工作方式如下。除了 $\mathbb{Z}$-向量加法系统图的原有计数器外，对于每个变量 $\mathrm{x}$，它还设有 $\in X$ 两个额外计数器：$\mathrm{x}^+$ 和 $\mathrm{x}^-$。这里，$\mathrm{x}^+$ （$\mathrm{x}^-$）用于计数 $\mathrm{x}$ 被赋予正（负）赋值的次数。通过确保最终 $\mathrm{x}^+ = 0$ 或 $\mathrm{x}^- = 0$，我们就能保证始终读取 $\mathrm{x}$ 的同一个值。

因此，为了检查一个文字，我们的 $\mathbb{Z}$-VASS 会递增相应的计数器。最后，在到达终态之前，它会依次处理每个变量 $\mathrm{x} \in X$，并进入一个递减 $\mathrm{x}^+$ 的循环，或者进入一个递减 $\mathrm{x}^-$的循环。这样，只有在所有变量检查都一致时，它才能到达零向量。

从 PRBCA 到 $\mathbb{Z}$-文法。接下来需要在对数空间内将一个等价于 $\mathbb{Z}$-文法的 $(r, k)$-PRBCA 转换为 $\mathbb{Z}$-文法。与将 RBCA 转换为 $\mathbb{Z}$-VASS 类似，可以将 PRBCA 转换为等价的 $\mathbb{Z}$-PVASS（带有 $\mathbb{Z}$-计数器的下推向量加法系统）。随后，应用从下推自动机到上下文无关文法的经典转换（又称三元组构造），参见 [1, 引理 2.26]：对于每个状态对 $(p, q)$，引入一个非终结符 $X_{p,q}$，它推导出所有在 $p$ 到 $q$ 之间读取的字符串（从空栈开始并以空栈结束）。例如，对于所有压入转移 $(p, a, \gamma, p')$ 和弹出转移 $(q', b, \bar{\gamma}, q)$，我们引入产生式 $X_{p,q} \to a X_{p',q'} b$。$\mathbb{Z}$-PVASS 中转移的计数器效应（$\mathbb{Z}^k$中的向量）会转换为产生式的效应。例如，上述产生式 $X_{p,q} \to a X_{p',q'} b$ 的效应，就是相应压入转移和弹出转移的效应之和。

## 4 判定无界性谓词

**证明概要。** 本节中，我们证明定理 2。先简要概述一下。我们的任务是获取一个 PRBCA $\mathcal{A}$，并非确定性地计算出一个

下推自动机 $\mathcal{A}'$，使得 $L(\mathcal{A})$ 满足 $\mathfrak{p}$，当且仅当 $\mathcal{A}'$ 的某些结果满足 $\mathfrak{p}$。从构造中可以清楚地看出，如果输入是 RBCA，那么得到的下推自动机将是非确定有限自动机。利用定理1，我们将把归约的主要部分表述为 $\mathbb{Z}$-文法，也就是说，以一个 $\mathbb{Z}$-文法 $G$ 为输入，并非确定性地计算出上下文无关文法 $G'$。

归约的思路是在 $G$ 中识别一组产生式，从某种适当的意义上说，这组产生式可以由另一组产生式抵消（就整数计数器值而言）。然后，$G'$ 仅使用一组能够被抵消的产生式得到。此外，无论这些产生式执行何种计数器更新，都会使用它们。接着，为了证明正确性，我们从两个方向进行论证：首先，我们证明任何可由 $G'$ 推导出的词，都是 $L(G)$ 的一个因子。从本质上说，这是因为 $G'$ 中使用的每个产生式，都可以通过在 $G$ 中添加更多产生式来抵消，从而得到 $G$ 的一个完全推导。因此，我们有 $L(G') \subseteq F(L(G))$；根据无界性谓词公理，这意味着 $\mathfrak{p}(L(G'))$ 蕴含 $\mathfrak{p}(L(G))$。其次，我们证明 $L(G)$ 是若干乘积（即连接）$P_i = L_1 \cdot L_2 \cdots L_k$ 的有限并，其中每个 $L_i$ 要么是有限的，要么包含于某个非确定性结果对应的 $L(G')$ 中。再次，根据无界性谓词公理，这意味着如果 $\mathfrak{p}(L(G))$，那么对于某个 $G'$，必有 $\mathfrak{p}(L(G'))$ 成立。

**无界性谓词与有限语言**。在开始证明之前，我们先注意到，可以假设我们的无界性谓词只对无限集合成立。首先，假设 $\mathfrak{p}$ 对 $\{\varepsilon\}$成立。这意味着 $\mathfrak{p} = \mathfrak{p}_{\neq\emptyset}$，因此我们只需通过判断 $L \neq \emptyset$，就能判定 $\mathfrak{p}(L)$，而这可以在 NP [33]中完成。从现在起，假设 $\mathfrak{p}$ 对 $\{\varepsilon\}$不成立。考虑字母表 $\Sigma_1 := \{a \in \Sigma \mid \mathfrak{p}(\{a\})\}$。现在注意到，如果 $K \subseteq \Sigma^*$ 是有限的，那么根据无界性谓词公理，$\mathfrak{p}(K)$当且仅当 $\Sigma_1$ 中的某个字母出现在 $K$中。因此，如果 $L \subseteq (\Sigma \setminus \Sigma_1)^*$，那么只有当 $L$ 是无限的，$\mathfrak{p}(L)$才可能成立。这引出了以下定义。给定一个语言 $L \subseteq \Sigma^*$，我们定义

$$L_0 = L \cap (\Sigma \setminus \Sigma_1)^*, \quad L_1 = L \cap \Sigma^* \Sigma_1 \Sigma^*.$$

那么，当且仅当 $\mathfrak{p}(L_0)$或 $\mathfrak{p}(L_1)$时，$\mathfrak{p}(L)$成立。此外，$\mathfrak{p}(L_1)$等价于 $L_1 \neq \emptyset$。

因此，我们的归约按如下方式进行。我们为 $L_0$和 $L_1$构造（P）RBCA。这可以在对数空间内完成，因为与正则语言求交可以通过简单的乘积构造实现。然后，我们在 NP 中检查 $L_1 \neq \emptyset$是否成立。如果成立，就返回“无界”。如果不成立，我们将 $\mathfrak{p}$ 视为定义在 $\Sigma \setminus \Sigma_1$ 上的语言的无界性谓词，并且它具有如下附加性质：$\mathfrak{p}$ 仅对无限语言成立。因此，只需在 $\mathfrak{p}$ 仅对无限集合成立的情况下证明定理 2。

**泵与消去**。为了定义可消去产生式的概念，我们需要先介绍一些术语。我们需要讨论 $\mathbb{Z}$-文法的推导树。对于任意字母表 $\Gamma$ 和 $d \in \mathbb{N}$，令 $\mathcal{T}_{\Gamma,d}$ 表示所有有限树的集合，其中每个节点都同时标记有 (i) 来自 $\Gamma$ 的一个字母，以及 (ii) 一个

来自 $\mathbb{Z}^d$ 的向量。设 $G = (N, \Sigma, P, S)$ 是一个 $d$ 维的 $\mathbb{Z}$-文法。对于产生式 $p = (A \to u, \boldsymbol{v})$，我们将其对应的计数器效应记为 $\varphi(p) := \boldsymbol{v}$。对于 $G$ 中的每个推导，我们都像上下文无关文法那样，将其关联到 $\mathcal{T}_{N\cup\Sigma,d}$ 中的一棵推导树。唯一的区别在于，每当我们应用产生式 $(A \to u, \boldsymbol{v})$ 时，与被改写的 $A$ 对应的节点也会标记为 $\boldsymbol{v}$。与上下文无关文法一样，叶节点携带终结符；它们的向量标记就是 $\mathbf{0} \in \mathbb{Z}^d$。

我们将映射 $\varphi$ 扩展到 $\mathbb{N}^P$ 中的两个向量以及推导树。如果 $\boldsymbol{u} \in \mathbb{N}^P$，则 $\varphi(\boldsymbol{u}) = \sum_{p\in P} \varphi(p)\cdot \boldsymbol{u}[p]$。类似地，如果 $\tau$ 是一棵推导树，那么 $\varphi(\tau) \in \mathbb{Z}^d$ 是 $\mathbb{Z}^d$ 中所有标签的总和。对于推导 $A \overset{*}{\Rightarrow} u$ 的推导树 $\tau$，如果满足 $A = S$、$u \in \Sigma^*$ 和 $\varphi(\tau) = \mathbf{0}$，则称其为完全推导。换言之，$\tau$ 推导出一个终结词，且该推导的计数器总效应为零。对于这样的完全推导，我们也用 $\mathrm{yield}(\tau)$ 表示词 $u$。如果推导树 $\tau$ 是某种形式为 $A \overset{*}{\Rightarrow} uAv$ 的推导的推导树，其中 $u$，$v \in \Sigma^*$ 且 $A \in N$，则称它为泵。如果存在一条 $G$ 的完全推导，其中包含集合 $M$ 中的所有非终结符，且不包含集合 $M$ 以外的任何非终结符，则称非终结符集合 $M \subseteq N$ 是可实现的。

如果存在泵 $\tau_1$、……、$\tau_k$（对于某个 $k \in \mathbb{N}$），使得 (i) $p$ 出现在某个 $\tau_i$ 中，且 (ii) $\varphi(\tau_1) + \cdots + \varphi(\tau_k) = \mathbf{0}$，也就是说，$\tau_1$、……、$\tau_k$ 的计数器总效应为零，并且 (iii) $\tau_1$、……、$\tau_k$ 中的所有产生式都只使用来自 $M$ 的非终结符，那么文法 $P$ 中的产生式 $p$ 就称为可消去的。如果子集 $Q \subseteq P$ 中的所有产生式都是 $M$-可消去的，我们就称 $Q \subseteq P$ 是 $M$ 可消去的。

**归约**。利用 $M$-可消去产生式的概念，我们现在可以描述如何构造上下文无关文法。假设 $M$ 是可实现的，$Q \subseteq P$ 是 $M$-可消去的，并且 $A \in M$。考虑语言

$$L_{A,Q} = \{u, v \in \Sigma^* \mid \exists \text{ derivation } A \overset{*}{\Rightarrow}_Q uAv\}.$$

因此，$L_{A,\,Q}$ 由所有在形式为 $A \overset{*}{\Rightarrow} uAv$ 的推导中出现的字符串 $u$ 和 $v$ 组成（其计数器值不一定为零），其中我们只使用 $M$-可消去产生式。$L_{A,\,Q}$ 将是上文提到的语言 $L(G')$。

容易观察到，给定 $G$ 和一个子集 $Q \subseteq P$，我们可以构造一个上下文无关文法，用于生成 $L_{A,Q}$：

**引理 1.** 给定一个 $\mathbb{Z}$-文法 $G$、一个非终结符 $A$ 和一个子集 $Q \subseteq P$，我们可以在对数空间内构造一个上下文无关文法，用于生成 $L_{A,Q}$。此外，如果 $G$ 是左线性的，那么该构造会生成一个用于 $L_{A,Q}$ 的非确定有限自动机。

我们在附录 C 中给出详细说明。现在，我们的归约如下进行：

1. 猜测一个子集 $M \subseteq N$ 和一个 $A \in M$；验证 $M$ 是否可实现。2. 猜测一个子集 $Q \subseteq P$；验证 $Q$ 是否可 $M$-消去。3. 计算一个用于生成 $L_{A,Q}$ 的上下文无关文法。

这里，我们需要证明步骤 1 和 2 可以在 NP 中完成：

**引理 2.** 给定一个子集 $M \subseteq N$，我们可以在 NP 中检验 $M$ 是否可实现。此外，给定 $M \subseteq N$ 和 $p \in P$，我们可以在 NP 中检验 $p$ 是否是 $M$-可消去的。

这两步都可以利用以下事实完成：对于给定的上下文无关文法，可以构造一个帕里克等价的存在预布尔公式 [56]，并且存在预布尔公式的可满足性问题属于 NP。详情请参见附录 C。至此，我们完成了对归约的描述。因此，接下来需要证明该归约的正确性。换言之，我们要证明：

**命题 1.** 当且仅当对于某个子集 $Q \subseteq P$，存在一个可实现的 $M \subseteq N$，使得 $A \in M$ 和 $Q$ 是 $M$- 可消去的时，才有 $\mathfrak{p}(L(G))$ $\mathfrak{p}(L_{A,Q})$。

命题 1 将通过两个引理来证明：

**引理 3.** 如果 $M$ 可实现且 $Q$ 是 $M$-可消去的，那么对于每个 $A \in M$，都有 $L_{A,Q} \subseteq F(L(G))$。

**引理 4.** $L(G)$ 包含于若干形如 $K_1 \cdot K_2 \cdots K_m$ 的集合的有限并中，其中每个 $K_i$ 要么是有限集，要么是一个集合 $L_{A,Q}$，其中对于某个可实现的 $M \subseteq N$，$Q$ 是 $M$-可消去的，并且 $A \in M$。

下面来看看命题 1 为什么可以由引理 3 和引理 4 推出。

证明（命题 *1*）。我们先证明“若”方向。设 $\mathfrak{p}(L_{A,Q})$ 对 $A$ 和 $Q$ 成立，具体如上所述。根据引理 3 以及无界性谓词的第一和第四条公理，可得 $\mathfrak{p}(L(G))$。

对于“仅若”方向，设 $\mathfrak{p}(L(G))$。根据无界性谓词的第一条公理，对于引理 4 给出的有限并，$\mathfrak{p}$ 必须成立。根据第二条公理，可得对于引理 4 中的某个有限乘积 $K_1 \cdots K_m$，$\mathfrak{p}(K_1 \cdots K_m)$ 成立。此外，根据第三条公理，可得对于某个 $i \in \{1, \ldots, m\}$，$\mathfrak{p}(K_i)$ 成立。如果 $K_i$ 是有限的，那么根据假设，$\mathfrak{p}(K_i)$ 不成立。因此，必有某个 $K_i = L_{A,Q}$ 使得 $\mathfrak{p}(K_i)$ 成立，证毕。□

**流。** 我们还需要证明引理 3 和引理 4。我们先从引理 3 开始，为此需要介绍一些术语。设 $\Sigma$ 是一个字母表。记 $\Psi\colon \Sigma^* \to \mathbb{N}^\Sigma$ 为帕里克映射，其定义为 $\Psi(w)(a) = |w|_a$，其中 $w \in \Sigma^*$ 和 $a \in \Sigma$。换句话说，$\Psi(w)(a)$ 是 $a$ 在 $w \in \Sigma^*$ 中出现的次数。如果 $\Gamma \subseteq \Sigma$ 是一个子集，那么 $\pi_\Gamma\colon \Sigma^* \to \Gamma^*$ 是满足 $\pi_\Gamma(a) = \varepsilon$（其中 $a \in \Sigma \setminus \Gamma$）以及 $\pi_\Gamma(a) = a$（其中 $a \in \Gamma$）的同态。我们还将 $\pi_\Gamma$ 称为到 $\Gamma$ 的投影。

假设有一个 $\mathbb{Z}$-文法 $G = (N, \Sigma, P, S)$，其非终结符为 $N$，产生式为 $P$。对于推导树 $\tau$，我们用 $\Psi(\tau)$ 表示 $\mathbb{N}^P$中的向量，该向量统计每个产生式在 $\tau$中出现的次数。我们引入一个映射 $\partial$，用于统计每个产生式消耗和产生的非终结符数量。形式上，$\partial\colon \mathbb{N}^P \to \mathbb{Z}^N$ 是一个幺半群同态，它将产生式$p = A \to w$映射为向量 $\partial(p) = -A + \Psi(\pi_N(w))$。这里，$-A \in \mathbb{Z}^N$表示一个向量，其在 $A$ 对应的位置上的值为 $-1$，其他位置上的值均为 0。若向量 $\boldsymbol{u} \in \mathbb{N}^P$ 满足 $\partial(\boldsymbol{u}) = \mathbf{0}$，则称其为流。注意，推导树 $\tau$是泵，当且仅当 $\Psi(\tau)$ 是一个流。在这种情况下，我们也将满足 $\boldsymbol{u} = \Psi(\tau)$ 的向量 $\boldsymbol{u} \in \mathbb{N}^P$称为泵。

下面的引理将提供一种构造推导的简便方法。这是 Esparza 的一个著名结果 [19,定理 3。1]，此后已被用于多个关于上下文无关文法的结果中。我们的表述比 Esparza 的表述略弱，但足以满足我们的目的，而且证明简单，其思路源自 Kufleitner 的一个证明 [45]。

**引理 5.** 设 $\boldsymbol{f} \in \mathbb{N}^P$。那么，$\boldsymbol{f}$ 是一个流，当且仅当它是若干泵之和。

证明. “若”方向显然成立，因为每个泵显然都是一个流。反过来，假设 $\boldsymbol{f} \in \mathbb{N}^P$是一个流。显然，我们可以写成 $\boldsymbol{f} = \Psi(\tau_1) + \cdots + \Psi(\tau_n)$，其中 $\tau_1, \ldots, \tau_n$ 是推导树：只需将 $\boldsymbol{f}$中的每个产生式都视为一棵独立的推导树。现在假设我们有 $\boldsymbol{f} = \Psi(\tau_1) + \cdots + \Psi(\tau_n)$，使得 $n$ 是最小的。我们声称，此时每个 $\tau_i$ 都是一个泵，从而证明了该引理。

假设并非如此，那么不失一般性地，$\tau_1$ 不是一个泵。由于 $\tau_1$ 是一个推导，这意味着 $\Psi(\tau_1)$不能是一个流，因此必定存在一个非终结符 $A$，使得 $\partial(\tau_1)(A) \neq 0$。

我们首先假设 $\partial(\tau_1)(A) > 0$。这意味着，$\tau_1$ 的某个叶节点上出现了一个非终结符$A$，且 $A$ 不是 $\tau_1$的开始符号。由于$\boldsymbol{f} = \Psi(\tau_1)+\cdots+\Psi(\tau_n)$是一个流，因此必有 $\partial(\Psi(\tau_2)+\cdots+\Psi(\tau_n))(A) < 0$。反过来，这只有在某个$\tau_j$ 以 $A$ 为开始符号时才可能。因此，我们可以将 $\tau_1$ 和$\tau_j$ 合并，方法是用新的子树$\tau_j$替换 $\tau_1$上带有 $A$标签的叶节点。由此得到一组新的 $n-1$ 棵树，其 Parikh 像为 $\boldsymbol{f}$，这与 $n$的选取相矛盾。若 $\partial(\tau_1)(A) < 0$，那么必定存在一个$\tau_j$，使得 $\partial(\tau_j)(A) > 0$，因而我们可以将 $\tau_1$ 插入$\tau_j$下方，从而得到类似的矛盾。□

**构造推导。** 利用流，我们现在可以证明引理 3。

证明。 假设存在一个推导 $\tau\colon A \overset{*}{\Rightarrow}_Q uAv$，满足 $A \in M$ 且 $u,\ v \in \Sigma^*$。我们必须证明，$u$ 和 $v$ 都出现在某个词 $w \in L(G)$ 中。此外，如果 $G$ 处于乔姆斯基范式，我们可以选择 $w$，使得 $|w|$ 关于 $|u|$和 $|v|$呈线性。我们的目标是构造一个 $G$ 的推导，其中 $u$ 和 $v$ 是因子。我们可以通过将 $\tau$ 插入某个 $G$ 的推导树（在 $A$的某次出现处）来获得一棵推导树，但这可能会产生非零的计数器值。因此，我们将利用 $Q$ 是 $M$-可消去的这一事实，找出其他也可以插入的泵，从而使计数器重新归零。

由于 $M \subseteq N$ 是可实现的，存在一个完全推导 $\tau_0$，它推导出某个词 $w_0 \in L(G)$，并且恰好使用 $M$中的非终结符。由于 $Q \subseteq P$是 $M$-可消去的，我们知道，对于每个产生式 $p \in Q$，都存在泵$\tau_1$、…、$\tau_k$，使得 (i) $p$出现在某个 $\tau_i$中，(ii) $\varphi(\tau_1) + \cdots + \varphi(\tau_k) = \mathbf{0}$，以及 (iii) $\tau_1$、…、$\tau_k$ 中的所有产生式都只使用 $M$中的非终结符。由此我们可以定义 $\boldsymbol{f}_p := \Psi(\tau_1) + \cdots + \Psi(\tau_k)$。注意，$\boldsymbol{f}_p$只包含含有来自 $M$ 的非终结符的产生式，因此有 $\boldsymbol{f}_p[p] > 0$，且 $\varphi(\boldsymbol{f}_p) = 0$。我们可以利用流 $\boldsymbol{f}_p$ 来找到所需的抵消泵。根据引理 5，每个流都可以分解为若干泵之和，因此只需构造一个特定的流。具体来说，我们要寻找一个流 $\boldsymbol{f}_\tau \in \mathbb{N}^P$，使得：

1. 任意产生式 $p$包含的 $\boldsymbol{f}_\tau[p] > 0$仅使用来自 $M$的非终结符，并且

2. $\varphi(\boldsymbol{f}_\tau + \Psi(\tau)) = 0$.

第一个条件确保所有得到的泵都可以插入 $\tau_0$。第二个条件确保得到的计数器总值为零。我们声称，借助

$$\boldsymbol{f}_\tau = \left( \sum_{p \in Q} \Psi(\tau)[p] \cdot \boldsymbol{f}_p \right) - \Psi(\tau), \tag{1}$$

我们达到了这些条件。首先，注意到 $\boldsymbol{f}_\tau \in \mathbb{N}^P$：我们有

$$\boldsymbol{f}_\tau[q] \quad \geq \quad \Psi(\tau)[q] \cdot \boldsymbol{f}_q[q] - \Psi(\tau)[q] \quad = \quad \Psi(\tau)[q] \cdot (\boldsymbol{f}_q[q] - 1)$$

由于根据定义 $\boldsymbol{f}_q[q]$ 必定非零，因此其至少为零。其次，注意到$\boldsymbol{f}_\tau$ 确实是一个流，因为它是流的一个 $\mathbb{Z}$-线性组合。此外，出现在 $\boldsymbol{f}_\tau$ 中的所有产生式也都出现在某个 $p \in Q$ 对应的 $\boldsymbol{f}_p$ 中，或出现在 $\tau$ 中，这意味着所有非终结符都必须属于 $M$。最后，$\boldsymbol{f}_\tau + \Psi(\tau)$的计数器总效应为零，因为 $\boldsymbol{f}_\tau + \Psi(\tau) = \sum_{p \in Q} \Psi(\tau)[p] \cdot \boldsymbol{f}_p$是若干流之和，而每个流的计数器总效应都为零。

现在，由于 $\boldsymbol{f}_\tau$是一个流，引理 5 告诉我们，存在泵 $\tau'_1$、.....、 $\tau'_m$，使得 $\boldsymbol{f}_\tau = \Psi(\tau'_1) + \cdots + \Psi(\tau'_m)$。因此，将 $\tau$和 $\tau'_1$、.....、 $\tau'_m$插入$\tau_0$ 中，必定会得到一个词的推导，该词同时以 $u$ 和 $v$ 为因子，并且其计数器值为

$$\underbrace{\varphi(\tau_0)}_{=\mathbf{0}} + \underbrace{\varphi(\tau) + \varphi(\tau'_1) + \cdots \varphi(\tau'_m)}_{=\varphi(\tau)+\varphi(\boldsymbol{f}_\tau)=\mathbf{0}} = \mathbf{0}.$$

因此，我们得到了 $G$ 的一个完全推导。因此 $L_{A,\ Q} \subseteq F(L(G))$. □

**有限并分解**。 接下来要证明引理 4。为进行分解，我们证明存在一个有限集 $D_0$，其中包含完全推导，使得 $G$ 的所有完全推导都可以通过从 $D_0$ 中某个推导出发，再插入能够生成 $L_{A,\ Q}$ 中字符串的泵来得到，其中 $A$ 和 $Q$ 适当。关键在于，集合 $D_0$ “基础推导”是有限的。对于上下文无关文法，只需像 Parikh定理 [52] 中那样，基于鸽巢原理进行简单的“反泵”论证即可证明这一点。然而，对于 $\mathbb{Z}$-文法， $D_0$ 只能包含计数器值为零的推导，因此这一点并不明显。为此，我们在（带标签的）树上使用良拟序。回顾一下，拟序 是自反且传递的序关系。对于一个拟序 $(X,\ \leq)$ 和一个子集 $Y \subseteq X$，我们将 $Y\uparrow$ 记作集合 $\{x \in X \mid \exists y \in Y\colon\ y \leq x\}$。如果每个非空子集 $Y \subseteq X$ 都有一个有限子集 $Y_0 \subseteq Y$ 使得 $Y \subseteq Y_0\uparrow$，我们就称 $(X,\ \leq)$ 是一个良拟序 （WQO）。

我们在 $\mathcal{T}_{N \cup \Sigma, d}$ 中所有的树上定义一种序关系。树 $s$ 是 $t$ 的一棵子树，当且仅当存在 $t$ 中的一个节点 $x$，使得 $s$ 由 $t$ 中所有作为 $x$ 后代的节点组成。如果 $\tau_1, \ldots, \tau_n$ 是树，那么我们用 $r[\tau_1, \ldots, \tau_n]$ 表示一棵树，其根节点为 $r$，子树 $\tau_1, \ldots, \tau_n$ 直接位于根节点之下。现在令 $\tau = (A, \boldsymbol{u})[\tau_1, \ldots, \tau_n]$

并且 $\tau' = (B, \boldsymbol{v})[\sigma_1, \ldots, \sigma_m]$ 是 $\mathcal{T}_{N\cup\Sigma,d}$ 中的树。我们如下定义序关系 $\preceq$。如果 $n = 0$（即 $\tau$ 仅由一个节点组成），那么当且仅当 $A = B$ 且 $m = 0$ 时，有 $\tau \preceq \tau'$。如果 $n \geq 1$，则我们递归地定义：

$$\tau \preceq \tau' \iff A = B \text{ and } \exists \text{ subtree } \tau'' = (A, \boldsymbol{u}')[\tau'_1, \ldots, \tau'_n] \text{ of } \tau' \text{ with } \tau_i \preceq \tau'_i \text{ for } i = 1, \ldots, n$$

基于 $\preceq$，我们定义如下的细微改进：当且仅当 $\tau \preceq \tau'$，并且出现在 $\tau$ 中的非终结符集合与出现在 $\tau'$ 中的集合相同时，我们写作 $\tau \sqsubseteq \tau'$。

**引理 6.** $(\mathcal{T}_{N\cup\Sigma,d}, \sqsubseteq)$ 是良拟序。

证明。在 [48, 引理 3.3], 中已证明 $\preceq$ 是良拟序。因此，$\sqsubseteq$ 是有限集上的相等关系与良拟序 $\preceq$ 的乘积。有限集上的相等关系是良拟序。□

引理 6 使我们能够将 $L(G)$ 分解为有限并：对于 $G$ 的每个完全推导 $\tau$，我们定义

$$L_\tau(G) = \{w \in \Sigma^* \mid \exists \text{完全推导 } \tau', \text{其中 } \tau \sqsubseteq \tau', \text{并产生 } \mathrm{yield}(\tau') = w\}.$$

**引理 7。** 存在一个有限集 $D_0 \subseteq \mathcal{T}_{N\cup\Sigma,\ d}$，其中包含 $G$的完全推导，满足 $L(G) = \bigcup_{\tau\in D_0} L_\tau(G)$。

证明。由于 $(\mathcal{T}_{N\cup\Sigma,\ d},\ \sqsubseteq)$ 是良拟序，$G$ 的所有完整推导构成的集合 $D \subseteq \mathcal{T}_{N\cup T,\ d}$ 有一个有限子集 $D_0$，满足 $D \subseteq D_0{\uparrow}$。这就证明了该引理。□

**分解为有限乘积。** 根据引理 7，接下来需要证明：对于每棵树 $\tau$，我们都能找到一个由语言构成的乘积 $K_1 \cdot K_2 \cdots K_m$，使得 $L_\tau(G) \subseteq K_1 \cdot K_2 \cdots K_m$，并且每个 $K_i$ 要么是有限的，要么具有 $L_{A,\ Q}$ 的形式。我们按如下方式归纳构造 $L_\tau(G)$ 的过近似。设 $M \subseteq N$ 和 $Q \subseteq P$ 分别是非终结符集和产生式集的子集。如果 $\tau$ 有一个节点，且该节点标记为 $a \in \Sigma$，那么我们令 $\mathrm{App}_Q(\tau) := \{a\}$。此外，如果$\tau = (A,\ \boldsymbol{u})[\tau_1, \ldots, \tau_n]$ 对于 $A \in N$ 和树 $\tau_1, \ldots, \tau_n$ 成立，那么我们令

$$\mathrm{App}_Q(\tau) := L_{A,Q} \cdot \mathrm{App}_Q(\tau_1) \cdot \mathrm{App}_Q(\tau_2) \cdots \mathrm{App}_Q(\tau_n) \cdot L_{A,Q}.$$

最后，我们设 $\mathrm{App}(\tau) := \mathrm{App}_Q(\tau)$，其中 $Q \subseteq P$是所有 $M$-可消去产生式的集合，$M$ 是所有出现在 $\tau$中的非终结符的集合。现在显然，每个 $\mathrm{App}(\tau)$ 都是一个有限乘积 $K_1 \cdot K_2 \cdots K_m$，正如所需：这是通过对 $\tau$ 的大小进行归纳得出的。因此，要证明引理 4，只需证明以下内容：

**引理 8。** 对于 $G$ 的每棵完整推导树 $\tau$，都有 $L_\tau(G) \subseteq \mathrm{App}(\tau)$。

证明。假设 $w \in L_\tau(G)$ 是通过带有 $\tau \sqsubseteq \tau'$ 的完整推导树 $\tau'$ 推导得到的。那么，$\tau$ 中出现的非终结符集合必须与 $\tau'$ 中的相同；我们将其记为 $M$。令 $Q \subseteq P$ 为所有 $M$-可消去产生式的集合。此外，由于 $\tau \preceq \tau'$，我们可以观察到存在泵 $\tau_1, \ldots, \tau_n$，其根非终结符为 $A_1, \ldots, A_n$，且在 $\tau$ 中的节点为 $x_1, \ldots, x_n$，使得将每个节点 $x_i$ 替换为泵 $\tau_i$，即可由 $\tau$ 得到 $\tau'$。

由于 $\tau$ 和 $\tau'$ 都是 $G$ 的完全推导，因此它们的计数器效应均为 $\mathbf{0}$。因此，$\varphi(\tau_1)+\cdots+\varphi(\tau_n) = \varphi(\tau')-\varphi(\tau) = \mathbf{0}$。于是，泵 $\tau_1,\ldots,\tau_n$ 表明，$\tau_1,\ldots,\tau_n$ 中出现的产生式是 $M$-可消去的。因此，与 $\tau_i$ 对应的推导只使用 $Q$ 中的产生式，所以 $\tau_i$ 对应于某个 $u_i, v_i$ 的 $A_i \Rightarrow_Q u_i A v_i$，并且有 $u_i, v_i \in L_{A,Q}$。□

## 5 增长

在本节中，我们证明定理 3。显然，有界语言具有多项式增长，因此只需证明：若 $L$ 由 PRBCA 接受且$L$ 无界，则其具有指数增长。对于两个语言 $L_1$，$L_2 \subseteq \Sigma^*$，我们记 $L_1 \hookrightarrow_{\text{lin}} L_2$，如果存在常数 $c \in \mathbb{N}$，使得对于每个词 $w_1 \in L_1$，都存在 $w_2 \in L_2$，满足 $|w_2| \leq c \cdot |w_1|$，且 $w_1$ 是 $w_2$的一个因子。不难看出，对于两个语言 $L_1$，$L_2 \subseteq \Sigma^*$，若 $L_1 \hookrightarrow_{\text{lin}} L_2$且 $L_1$ 具有指数增长，则 $L_2$也具有指数增长。

为了证明定理 3，我们需要引理 3 的一个改编版本。若一个$\mathbb{Z}$-文法的所有产生式都具有形式 $(A \to BC,\ \boldsymbol{v})$ 或 $(A \to a,\ \boldsymbol{v})$，其中 $A$，$B$，$C \in N$、$a \in \Sigma$，且 $\boldsymbol{u}$，$\boldsymbol{v} \in \mathbb{Z}^k$，则称该文法处于乔姆斯基范式 。换言之，忽略所有计数器向量后得到的上下文无关文法处于乔姆斯基范式。Fernau 和 Stiebe [21, 命题 5.12] 证明了，每个 $\mathbb{Z}$-文法都有一个等价的 $\mathbb{Z}$-文法处于乔姆斯基范式。

**引理 9**。 如果 $G = (N, \Sigma, P, S)$ 是乔姆斯基范式下的 $\mathbb{Z}$-文法，$M \subseteq N$可实现，$Q \subseteq P$是 $M$-可消去的，并且 $A \in M$，那么 $L_{A,\,Q} \hookrightarrow_{\text{lin}} L(G)$。

证明方式与引理 3 基本相同。现在我们证明，若某个 PRBCA 接受的语言 $L$ 不是有界的，那么它必定具有指数增长。上文已经看到，作为 PRBCA 语言，$L$ 由某个 $\mathbb{Z}$-文法生成。正如 Fernau 和 Stiebe [21, 命题 5.12], 所示，这意味着对于某个乔姆斯基范式下的 $\mathbb{Z}$-文法 $G$，有 $L = L(G)$。由于$L$ 无界，引理 4 得出 $A$ 和 $Q$，使得 $L_{A,\,Q}$ 不是有界语言。众所周知，任何非有界的上下文无关语言都具有指数增长（这一事实显然至少被独立发现过六次，参考文献见 [25] ）。因此，$L_{A,\,Q}$ 具有指数增长。由引理 9 可得 $L_{A,\,Q} \hookrightarrow_{\text{lin}} L$，因此 $L$ 具有指数增长。

**致谢** 我们感谢 Manfred Kufleitner 在手稿 [45] 公开发表之前与我们分享。该手稿为构造上下文无关文法的 Parikh 像的存在型 Presburger 公式提供了另一种证明。 [19, 56]中也证明了后者。我们在引理 5 中使用这一结果；引理 5 也可以由 [19, 定理 3.1]推导得出。不过，受 Kufleitner 的证明启发，我们为引理 5 给出了一个简单直接的证明。

本研究由欧盟资助（欧洲研究委员会，FINABIS，101077902）。然而，文中观点和意见仅代表作者本人，并不一定反映欧盟或欧洲研究委员会执行局的立场。欧盟及资助机构均不对此承担责任。

## 参考文献

[1] Alfred V. Aho 和 Jeffrey D. Ullman。《语法分析、翻译与编译理论》。第 1 卷：语法分析。Prentice-Hall，1972。isbn：0139145567。url：https://www.worldcat.org/oclc/310805937。[2] Brenda S Baker 和 Ronald V Book。“反转有界多推栈机”。发表于：《计算机与系统科学杂志》8.3（1974），第 315–332 页。doi：10.1016/S0022-0000(74)80027-9。[3] David Barozzini、Lorenzo Clemente、Thomas Colcombet 和 Pawel Parys。“代价自动机、安全方案与向下闭包”。发表于：第 47 届自动机、语言与编程国际研讨会（ICALP 2020），2020 年 7 月 8–11 日，德国萨尔布吕肯（线上会议）。编者：Artur Czumaj、Anuj Dawar 和 Emanuela Merelli。第 168 卷。LIPIcs。Dagstuhl城堡—莱布尼茨信息学中心，2020，109:1–109:18。doi：10.4230/LIPIcs.ICALP.2020.109。[4] David Barozzini、Pawel Parys 和 Jan Wroblewski。“递归方案的无界性：更简单的类型系统”。发表于：第 49 届自动机、语言与编程国际研讨会（ICALP 2022），2022 年 7 月 4–8 日，法国巴黎。编者：Mikolaj Bojanczyk、Emanuela Merelli 和 David P. Woodruff。第 229 卷。LIPIcs。Dagstuhl城堡—莱布尼茨信息学中心，2022，112:1–112:19。doi：10.4230/LIPIcs.ICALP.2022.112。[5] Alin Bostan、Arnaud Carayol、Florent Koechlin 和 Cyril Nicaud。“弱无歧义 Parikh 自动机及其与全纯级数的联系”。发表于：第 47 届自动机、语言与编程国际研讨会（ICALP 2020），2020 年 7 月 8–11 日，德国萨尔布吕肯（线上会议）。第 168 卷。LIPIcs。Dagstuhl 城堡 — 莱布尼茨信息学中心，2020，114:1–114:16。doi：10.4230/LIPIcs.ICALP.2020.114。[6] Toby Cathcart Burn、Luke Ong、Steven J. Ramsay 和 Dominik Wagner。“初始极限 Datalog：一类新的可扩展可判定约束 Horn 子句”。发表于：第 36 届 ACM/IEEE 逻辑与计算机科学年度研讨会（LICS 2021），意大利罗马，2021 年 6 月 29 日至 7 月 2 日。IEEE，2021，第 1–13 页。doi：10.1109/LICS52264.2021.9470527。[7] Michaël Cadilhac、Alain Finkel 和 Pierre McKenzie。“仿射 Parikh 自动机”。发表于：RAIRO Theor. Informatics Appl. 46.4（2012），第 511–545 页。doi：10.1051/ita/2012013。[8] Michaël Cadilhac、Alain Finkel 和 Pierre McKenzie。“有界 Parikh 自动机”。发表于：Int. J. Found. Comput. Sci. 23.8（2012），第 1691–1710 页。doi：10.1142/S0129054112400709。[9] Michaël Cadilhac、Andreas Krebs 和 Pierre McKenzie。“Parikh 自动机的代数理论”。发表于：Theory Comput. Syst. 62.5（2018），第 1241

–

1268. DOI：10.1007/s00224-017-9817-2。[10] Arturo Carpi、Flavio D’Alessandro、Oscar H Ibarra 和 Ian McQuillan。“有界语言、计数器机、有限指标文法、歧义性与交换正则性之间的关系”。载于：理论计算机科学862 (2021)，第 97–118 页。DOI：10.1016/j.tcs.2020.10.006。

[11] Tat-hung Chan。“具有反转有界计数器的下推自动机”。载于：计算机与系统科学杂志 37.3 (1988)，第 269–291 页。doi：10.1016/0022-0000(88)90008-6.[12] Laura Ciobanu 和 Susan Hermiller。“群中的共轭增长级数与语言”。载于：美国数学会会刊 366.5 (2014)，第 2803–2825 页。[13] Lorenzo Clemente、Wojciech Czerwinski、Slawomir Lasota 和 Charles Paperman。“Parikh 自动机的正则可分性”。载于：第44届自动机、语言与程序设计国际学术会议（ICALP 2017），2017年7月10日至14日，波兰华沙。第80卷。LIPIcs。Dagstuhl城堡—莱布尼茨信息学中心，2017，117:1–117:13。doi：10.4230/LIPIcs.ICALP.2017.117。[14] Lorenzo Clemente、Pawel Parys、Sylvain Salvati 和 Igor Walukiewicz。“高阶递归方案的对角问题可判定”。载于：第31届 ACM/IEEE 计算机科学逻辑年度研讨会论文集（LICS ’16），美国纽约州纽约市，2016年7月5日至8日。Martin Grohe、Eric Koskinen 和 Natarajan Shankar 编。ACM，2016，第 96–105 页。doi：10.1145/2933575.2934527。[15] Wojciech Czerwinski、Piotr Hofman 和 Georg Zetzsche。“向量加法系统语言的无界性问题”。载于：第45届自动机、语言与程序设计国际学术会议（ICALP 2018），2018年7月9日至13日，捷克共和国布拉格。Ioannis Chatzigiannakis、Christos Kaklamanis、Dániel Marx 和 Donald Sannella 编。第107卷。LIPIcs。Dagstuhl城堡—莱布尼茨信息学中心，2018，119:1–119:15。doi：10.4230/LIPIcs.ICALP.2018.119。[16] Flavio D’Alessandro 和 Benedetto Intrigila。“有界上下文无关语言与正则语言的交换等价性：半线性情形”。载于：理论计算机科学 572 (2015)，第 1–24 页。doi：10.1016/j.tcs.2015.01.008。网址：https://doi.org/10.1016/j.tcs.2015.01.008。[17] Joey Eremondi、Oscar H. Ibarra 和 Ian McQuillan。“上下文无关语言与计数器语言的密度”。载于：语言理论进展——第19届国际会议 DLT 2015，英国利物浦，2015年7月27日至30日，论文集。Igor Potapov 编。第9168卷。计算机科学讲义。施普林格，2015，第 228–239 页。doi：10.1007/978-3-319-21500-6 18。[18] Joey Eremondi、Oscar H. Ibarra 和 Ian McQuillan。“上下文无关语言与计数器语言的密度”。载于：国际计算机科学基础杂志 29.2 (2018)，第 233–250 页。doi：10.1142/S0129054118400051。[19] Javier Esparza。“Petri 网、交换上下文无关文法与基本并行进程”。载于：Fundamenta Informaticae 31.1 (1997)，第 13–25 页。[20] Javier Esparza、Pierre Ganty 和 Rupak Majumdar。“有界验证的完美模型”。载于：第27届 IEEE 计算机科学逻辑年度研讨会论文集（LICS 2012），克罗地亚杜布罗夫尼克，2012年6月25日至28日。IEEE 计算机学会，2012，第 285–294 页。doi：10.1109/LICS.2012.39。

[21] Henning Fernau 和 Ralf Stiebe. “序列文法与带价自动机”。载于：理论计算机科学 276.1-2（2002），第 377–405 页。doi: 10.1016/S0304-3975(01)00282-1.[22] Emmanuel Filiot、Shibashis Guha 和 Nicolas Mazzocchi. “双向 Parikh 自动机”。载于：第 39 届 IARCS 软技术基础与理论计算机科学年会（FSTTCS 2019），2019 年 12 月 11–13 日，印度孟买。第 150 卷。LIPIcs。Dagstuhl城堡—莱布尼茨信息学中心，2019，40:1–40:14。doi: 10.4230/LIPIcs. FSTTCS.2019.40.[23] Alain Finkel 和 Arnaud Sangnier. “重新审视反转有界计数器机器”。载于：2008 年计算机科学数学基础，第 33 届国际研讨会（MFCS 2008），波兰托伦，2008 年 8 月 25–29 日，论文集。Edward Ochmanski 和 Jerzy Tyszkiewicz 编。第 5162 卷。计算机科学讲义。施普林格，2008，第 323–334 页。doi: 10.1007/978-3-540-85238-4_26.[24] Moses Ganardi. “滑动窗口模型中的语言识别”。博士论文。锡根大学，德国，2019。url: https://dspace.ub.uni-siegen.de/handle/ubsi/1523.[25] Pawel Gawrychowski、Dalia Krieger、Narad Rampersad 和 Jeffrey O. Shallit. “在多项式时间内求正则语言或上下文无关语言的增长率”。载于：国际计算机科学基础杂志 21.4（2010），第 597–618 页。doi: 10.1142/S0129054110007441.[26] Seymour Ginsburg 和 Edwin H Spanier. “有界 ALGOL 类语言”。载于：美国数学会会刊 113.2（1964），第 333–368 页。[27] Sheila A. Greibach. “关于盲式和部分盲式单向多计数器机器的评述”。载于：理论计算机科学 7（1978），第 311–324 页。doi: 10.1016/0304-3975(78)90020-8.[28] Rostislav Grigorchuk 和 A. Machì. “一个具有中间增长的索引语言示例”。载于：理论计算机科学 215.1-2（1999），第 325–327 页。[29] Eitan M. Gurari 和 Oscar H. Ibarra. “有限转向多计数器机器判定问题的复杂性”。载于：计算机与系统科学杂志 22.2（1981），第 220–229 页。issn: 0022-0000。doi: https: // doi.org/10.1016/0022-0000(81)90028-3.[30] Christoph Haase 和 Simon Halfon. “带状态整数向量加法系统”。载于：可达性问题——第 8 届国际研讨会（RP 2014），英国牛津，2014 年 9 月 22–24 日，论文集。Joël Ouaknine、Igor Potapov 和 James Worrell 编。第 8762 卷。计算机科学讲义。施普林格，2014，第 112–124 页。doi: 10.1007/978-3-319-11439-2_9.[31] Christoph Haase 和 Georg Zetzsche. “带星号的 Presburger 算术、图群的有理子集与嵌套零测试”。载于：第 34 届 ACM/IEEE 计算机科学逻辑研讨会（LICS 2019），加拿大不列颠哥伦比亚省温哥华，2019 年 6 月 24–27 日。IEEE，2019，第 1–14 页。doi:

10.1109/LICS.2019.8785850。url: https://doi.org/10.1109/LICS.2019.8785850.[32] Matthew Hague、Jonathan Kochems 和 C.-H. Luke Ong. “高阶下推自动机的无界性与向下闭包”。载于：第 43 届 ACM SIGPLAN-SIGACT 编程语言原理研讨会论文集（POPL 2016），美国佛罗里达州圣彼得堡，2016 年 1 月 20–22 日。ACM，2016，第 151–163 页。doi: 10 . 1145 / 2837614.2837627.[33] Matthew Hague 和 Anthony Widjaja Lin. “使用数值数据类型对递归程序进行模型检验”。载于：计算机辅助验证——第 23 届国际会议（CAV 2011），美国犹他州 Snowbird，2011 年 7 月 14–20 日，论文集。Ganesh Gopalakrishnan 和 Shaz Qadeer 编。第 6806 卷。计算机科学讲义。施普林格，2011，第 743–759 页。doi: 10.1007/978-3-642-22110-1_60.[34] Simon Halfon、Philippe Schnoebelen 和 Georg Zetzsche. “子词序上一阶逻辑的可判定性、复杂性与表达能力”。载于：第 32 届 ACM/IEEE 计算机科学逻辑研讨会（LICS 2017），冰岛雷克雅未克，2017 年 6 月 20–23 日。IEEE 计算机学会，2017，第 1–12 页。doi: 10.1109/LICS.2017.8005141.[35] Takeshi Hayashi. “论索引文法的推导树——uvwxy 定理的扩展”。载于：数学科学研究所出版物 9.1（1973），第 61–92 页。doi: 10 . 2977 / prims / 1195192738.[36] John E. Hopcroft. “上下文无关语言的等价性与包含问题”。载于：数学系统理论 3.2（1969），第 119–124 页。[37] Oscar H. Ibarra. “反转有界多计数器机及其判定问题”。载于：ACM 杂志 25.1（1978），第 116–133 页。doi: 10 . 1145 / 322047.322058.[38] Oscar H. Ibarra 和 Bala Ravikumar. “关于接受器与转换器的稀疏性、歧义性及其他判定问题”。载于：STACS 86，第 3 届计算机科学理论方面年会，法国奥尔赛，1986 年 1 月 16–18 日，论文集。Burkhard Monien 和 Guy Vidal-Naquet 编。第 210 卷。计算机科学讲义。施普林格，1986，第 171–179 页。doi: 10.1007/3-540-16078-7_74.[39] Oscar H. Ibarra 和 Shinnosuke Seki. “用单向和双向确定性机器刻画有界半线性语言”。载于：国际计算机科学基础杂志 23.6（2012），第 1291–1306 页。doi: 10. 1142/S0129054112400539.[40] Oscar H. Ibarra、Jianwen Su、Zhe Dang、Tevfik Bultan 和 Richard A. Kemmerer. “计数器机器与验证问题”。载于：理论计算机科学 289.1（2002），第 165–189 页。doi: 10.1016/S0304-3975(01)00268-7.[41] Matthias Jantzen 和 Alexy Kurganskyy. “细化盲多计数器语言层级与扭转闭合三元组”。载于：信息与计算 185.2 （ 2003 ），第 159–181 页。doi: 10.1016/S0890-5401(03)00087-7。

[42] Felix Klaedtke 和 Harald Rueß。“带基数的单子二阶逻辑”。载于：*ICALP 2003* 论文集。由 Jos C. M. Baeten、Jan Karel Lenstra、Joachim Parrow 和 Gerhard J. Woeginger 编。柏林，海德堡：施普林格，2003 年，第 681–696 页。[43] NaokiKobayashi。“非确定性机器的前沿语言之间的包含关系 ic

递归程序方案与 Dyck语言是不可判定的”。载于：《理论计算机科学》777（2019），第409–416页。doi：10.1016/j.tcs.2018.09.035。[44] S. Rao Kosaraju。“向量加法系统中可达性的可判定性（初步版本）”。载于：STOC 1982，1982年5月5日至7日，美国加利福尼亚州旧金山。1982年，第267–281页。[45] Manfred Kufleitner。Parikh定理的又一个证明。2022年10月6日。arXiv：2210.02925。[46] Dietrich Kuske 和 Georg Zetzsche。“按子词序排列的语言”。载于：《软件科学与计算结构基础——第22届国际会议，FOSSACS 2019，作为欧洲软件理论与实践联合会议 ETAPS 2019 的一部分于捷克共和国布拉格举行，2019年4月6日至11日，论文集》。Mikolaj Bojanczyk 和 Alex Simpson 编。第11425卷。计算机科学讲义。施普林格，2019年，第348–364页。doi：10.1007/978-3-030-17127-8_20。[47] Jean-Luc Lambert。“判定 Petri 网可达性的一种结构”。载于：《理论计算机科学》99.1（1992），第79–104页。[48] Jérôme Leroux、M. Praveen、Philippe Schnoebelen 和 Grégoire Sutre。“关于下推Petri网及相关系统弱可计算的函数”。载于：CoRR abs/1904.04090（2019）。arXiv：1904.04090。[49] Jérôme Leroux 和 Sylvain Schmitz。“揭开向量加法系统可达性的奥秘”。载于：第30届 ACM/IEEE 计算机科学逻辑研讨会年会，LICS 2015，日本京都，2015年7月6日至10日。IEEE 计算机学会，2015年，第56–67页。doi：10.1109/LICS.2015.16。[50] Jérôme Leroux 和 Sylvain Schmitz。“固定维度下向量加法系统的可达性为原始递归复杂度”。载于：第34届 ACM/IEEE 计算机科学逻辑研讨会年会，LICS 2019，加拿大不列颠哥伦比亚省温哥华，2019年6月24日至27日。IEEE，2019年，第1–13页。doi：10.1109/LICS.2019.8785796。[51] Ernst W. Mayr。“一般Petri网可达性问题的算法”。载于：STOC 1981，1981年5月11日至13日，美国威斯康星州密尔沃基。1981年，第238–246页。[52] Rohit J Parikh。“论上下文无关语言”。载于：《美国计算机学会会刊》13.4（1966），第570–581页。[53] Pawel Parys。“递归 схем 对角问题的复杂性”。载于：第37届 IARCS 软件技术与理论计算机科学基础年会，FSTTCS 2017，2017年12月11日至15日，印度坎普尔。Satya V. Lokam 和 R. Ramanujam 编。第93卷。LIPIcs。Dagstuhl城堡—莱布尼茨信息学中心，2017年，45:1–45:14。doi：10.4230/LIPIcs.FSTTCS.2017.45。

[54] Loic Pottier。“线性丢番图系统的最小解：界与算法”。载于：重写技术与应用，第四届国际会议 RTA-91，意大利科莫，1991年4月10日至12日，论文集。Ronald V. Book 编。第488卷。计算机科学讲义。施普林格，1991年，第162–173页。doi：10.1007/3-540-53904-2_94。[55] George S. Sacerdote 和 Richard L. Tenney。“向量加法系统可达性问题的可判定性（初步版本）”。载于：STOC 1977。ACM。1977年，第61–76页。[56] Kumar Neeraj Verma、Helmut Seidl 和 Thomas Schwentick。“等式 Horn 子句的复杂性”。载于：自动演绎——CADE-20，第二十届自动演绎国际会议，爱沙尼亚塔林，2005年7月22日至27日，论文集。Robert Nieuwenhuis 编。第3632卷。计算机科学讲义。施普林格，2005年，第337–352页。doi：10.1007/11532231_25。[57] Georg Zetzsche。“计算向下闭包的一种方法”。载于：自动机、语言与编程——第42届国际研讨会，ICALP 2015，日本京都，2015年7月6日至10日，论文集，第二部分。Magnús M. Halldórsson、Kazuo Iwama、Naoki Kobayashi 和 Bettina Speckmann 编。第9135卷。计算机科学讲义。施普林格，2015年，第440–451页。doi：10.1007/978-3-662-47666-6_35。[58] Georg Zetzsche。“计算向下闭包的一种方法”。载于：CoRR abs/1503.01068（2015）。arXiv：1503.01068。[59] Georg Zetzsche。“向下闭包比较的复杂性”。载于：第43届自动机、语言与编程国际研讨会，ICALP 2016，2016年7月11日至15日，意大利罗马。Ioannis Chatzigiannakis、Michael Mitzenmacher、Yuval Rabani 和 Davide Sangiorgi 编。第55卷。LIPIcs。Dagstuhl城堡—莱布尼茨信息学中心，2016年，123:1–123:14。doi：10.4230/LIPIcs.ICALP.2016.123。

# A 第 2 节的补充材料

## A.1 计数器更新

这里我们说明，对于 $d$ 维的 $\mathbb{Z}$-向量加法系统，不失一般性，可以假设计数器更新采用一元编码。具体而言，我们证明，由 $n$ 位二进制编码的数构成的计数器更新，可以由由一元编码的数构成的计数器更新来模拟。此外，我们还提及如何将这一结果推广到 $\mathbb{Z}$-文法的情形，其使用的构造非常相似。总体思路是将每个计数器替换为 $n$ 个计数器，并将计数器更新中的每个数替换为其二进制表示。

设 $\mathcal{V} = (Q, \Sigma, q_0, T, F)$ 是一个 $d$ 维的 $\mathbb{Z}$-向量加法系统，其计数器更新采用二进制表示，每个数字最多占用 $n$ 位。我们按如下方式构造一个 $dn$ 维的$\mathbb{Z}$-向量加法系统 $\mathcal{V}' = (Q, \Sigma, q_0, T', F)$，其计数器更新采用一元编码：

– 对每个转移 $(p, w, (v_1, \ldots, v_d)\ p') \in T$，将转移 $(p, w, (\mathsf{bin}(v_1) \ldots, \mathsf{bin}(v_d))\ p')$ 添加到 $T'$ 中，其中 $\mathsf{bin} : \mathbb{Z} \to \{-1, 0, 1\}$ 将一个$n$位数字映射为其二进制表示，同时保留其符号。– 对每个 $i \in [1, d]$、每个 $j \in [1, n-1]$ 以及每个状态 $p \in Q$，将循环 $(p, \varepsilon, (u_1, \ldots, u_{dn})\ p)$ 和 $(p, \varepsilon, (v_1, \ldots, v_{dn})\ p)$ 添加到 $T'$ 中，其中 $u_{(i-1)n+j} = -2$、$u_{(i-1)n+j+1} = 1$、$v_{(i-1)n+j} = 2$、$v_{(i-1)n+j+1} = -1$ 和 $u_k = v_k = 0$ 对每个$k \notin \{(i-1)n + j, (i-1)n + j + 1\}$。

这里，第一组转移将二进制表示作为计数器效应，而第二组循环则确保模拟一个 $\mathcal{V}$计数器的每组 $n$ 计数器之间具有正确的关系。具体而言，这些循环保证每个这样的计数器都表示 $n$ 位中的一位，且每个后续位的权重都是前一位的两倍。

现在我们简要介绍一下 $\mathbb{Z}$-文法的构造。思路相同，但我们不替换转移，而是替换产生式。随后，我们不添加循环，而是为每个非终结符添加 $A$ 产生式，将$A$ 替换为其自身。在这两种情况下，计数器更新都与$\mathbb{Z}$-VASS构造中的完全相同，证明也非常相似。

为了证明 $\mathbb{Z}$-VASS 情形下的构造正确，我们对运行长度进行归纳，证明存在一个运行 $(q, u_1, \ldots, u_n) \xrightarrow{w} (p, v_1, \ldots, v_n)$，其对应于 $\mathcal{V}$，当且仅当存在一个运行 $(q, u'_1, \ldots, u'_{dn}) \xrightarrow{w} (p, v'_1, \ldots, v'_{dn})$，其对应于 $\mathcal{V}'$，且满足 $u_i = \sum_{j=1}^{n} 2^{j-1} u'_{(i-1)n+j}$ 和 $v_i = \sum_{j=1}^{n} 2^{j-1} v'_{(i-1)n+j}$。对于两种 $\mathbb{Z}$-VASS，初始配置给出的基例显然成立。

对于归纳情形，我们首先假设存在 $\mathcal{V}$ 的一次运行 $(q, u_1, \ldots, u_n) \xrightarrow{w} (p, v_1, \ldots, v_n) \xrightarrow{w'} (r, x_1, \ldots, x_n)$，其中第二部分由单次转移 $(p, w', \boldsymbol{x} - \boldsymbol{v}, r) \in T$ 得到。于是根据归纳假设，存在 $\mathcal{V}'$ 的一次运行 $(q, u'_1, \ldots, u'_{dn}) \xrightarrow{w} (p, v'_1, \ldots, v'_{dn})$，满足 $u_i = \sum_{j=1}^{n} 2^{j-1} u'_{(i-1)n+j}$ 和 $v_i = \sum_{j=1}^{n} 2^{j-1} v'_{(i-1)n+j}$。此外，根据构造，我们有 $(p, w', (\mathsf{bin}(x_1 - v_1) \ldots, \mathsf{bin}(x_d - v_d))\ r) \in T'$。将这次转移接到以 $(p, \boldsymbol{v}')$ 结束的 $\mathcal{V}'$ 的运行之后，得到配置 $(r, y'_1, \ldots, y'_{dn})$，其中对于 $i \in [1, d]$ 和 $j \in [1, n]$，$y'_{(i-1)n+j} = v'_{(i-1)n+j} + \mathsf{bin}(x_i - v_i)_j$。这里，$\mathsf{bin}(-)_j$ 指的是第 $j$ 位数字，

它是二进制表示中与 2 的 $(j-1)$ 次幂相对应的数字。这便得到 $\boldsymbol{x}$ 和 $\boldsymbol{y}'$ 之间如下所示的关系，符合要求：

$$\begin{aligned}\sum_{j=1}^{n} 2^{j-1} y'_{(i-1)n+j} &= \sum_{j=1}^{n} 2^{j-1} v'_{(i-1)n+j} + 2^{j-1}\mathsf{bin}(x_i - v_i)_j \\ &= (x_i - v_i) + \sum_{j=1}^{n} 2^{j-1} v'_{(i-1)n+j} = (x_i - v_i) + v_i = x_i.\end{aligned}$$

其次，现在假设存在 $\mathcal{V}'$ 的一次运行 $(q, u'_1, \ldots, u'_{dn}) \xrightarrow{w} (p, v'_1, \ldots, v'_{dn}) \xrightarrow{w'} (r, y'_1, \ldots, y'_{dn})$，其中第二部分由单次转移 $t' = (p, w', \boldsymbol{y}' - \boldsymbol{v}', r) \in T'$ 导致。根据归纳假设，存在 $\mathcal{V}$ 的一次运行 $(q, u_1, \ldots, u_n) \xrightarrow{w} (p, v_1, \ldots, v_n)$，满足 $u_i = \sum_{j=1}^{n} 2^{j-1} u'_{(i-1)n+j}$ 和 $v_i = \sum_{j=1}^{n} 2^{j-1} v'_{(i-1)n+j}$。现在需要区分两种情况：

– 如果根据转移 $t \in T$ 将 $t'$ 添加到 $T'$，那么根据构造，我们有 $t = (p, w', \left(\sum_{j=1}^{n} 2^{j-1}(y'_j - v'_j) \ldots, \sum_{j=1}^{n} 2^{j-1}(y'_{(d-1)n+j} - v'_{(d-1)n+j})\right), r)$。将此转移追加到以 $(p, \boldsymbol{v})$ 结束的 $\mathcal{V}$ 的运行中，会得到配置 $(r, x_1, \ldots, x_d)$，其中对于 $i \in [1, d]$，$x_i = v_i + \sum_{j=1}^{n} 2^{j-1}(y'_{(i-1)n+j} - v'_{(i-1)n+j})$。由此可得 $\boldsymbol{x}$ 和 $\boldsymbol{y}'$ 之间的以下关系，符合要求：

$$\begin{aligned}x_i &= v_i + \sum_{j=1}^{n} 2^{j-1}(y'_{(i-1)n+j} - v'_{(i-1)n+j}) \\ &= (v_i - v_i) + \sum_{j=1}^{n} 2^{j-1} y'_{(i-1)n+j} = \sum_{j=1}^{n} 2^{j-1} y'_{(i-1)n+j}.\end{aligned}$$

– 如果 $t'$ 是 $T'$ 中未基于 $T$ 的一次转移的循环之一，那么我们有 $w' = \varepsilon$、$p = r$，并且计数器效应 $\boldsymbol{y}' - \boldsymbol{v}'$ 将值 $v'_{(i-1)n+j}$ 改变 2，将值 $v'_{(i-1)n+j+1}$ 改变 1，其中 $i \in [1, d]$、$j \in [1, n-1]$ 为某些值。由于此效应不会改变其他值，它只影响总和 $\sum_{j=1}^{n} 2^{j-1} v'_{(i-1)n+j}$。由于变化 2 和变化 1 的符号总是相反的，因此该总和也不受影响。因此，以 $(p, \boldsymbol{v})$ 结束的 $\mathcal{V}$ 运行已经满足要求。

为了证明 $\mathcal{V}$ 和 $\mathcal{V}'$ 之间的语言等价，我们仍需证明：如果 $\mathcal{V}$ 到达 $\mathbf{0}$，那么 $\mathcal{V}'$ 也可以通过恰好到达 $\mathbf{0}$ 来模拟这一点。形式化地说，我们证明：$\mathcal{V}$ 存在一条运行 $(q_0, \mathbf{0}) \xrightarrow{w} (p, \mathbf{0})$，当且仅当 $\mathcal{V}'$ 存在一条运行 $(q_0, \mathbf{0}) \xrightarrow{w} (p, \mathbf{0})$。

“仅当”方向直接由我们刚刚证明的命题得出。对于“如果”方向，我们可知存在一条运行 $(q_0, \mathbf{0}) \xrightarrow{w} (p, \boldsymbol{v}')$，其属于 $\mathcal{V}'$ with $\sum_{j=1}^{n} 2^{j-1} v'_{(i-1)n+j} = 0$，对每个 $i \in [1, d]$ 均成立。假设对于某个 $i \in [1, d]$，存在最大的 $j \in [1, n]$ 使得 $v'_{(i-1)n+j} \neq 0$。那么 $j > 1$，因为上述总和必须等于 0，所以必定存在一个更小的 $j' \in [1, n]$，使得 $v'_{(i-1)n+j'} \neq 0$

也成立。考虑循环 $\ell = (p,\ \varepsilon,\ (u_1,\ \ldots\ldots,\ u_{dn})\ p) \in T'$，其中 $u_{(i-1)n+j-1} = -2$，若 $v_{(i-1)n+j} < 0$ 则为 $u_{(i-1)n+j} = 1$，或者若 $v_{(i-1)n+j} < 0$ 则为 $u_{(i-1)n+j-1} = 2,\ u_{(i-1)n+j} = -1$。将 $\ell$ $|v_{(i-1)n+j}|$ 多次附加到以 $(p,\ \boldsymbol{v}')$ 结束的运行中，会使值 $v_{(i-1)n+j}$ 变为 0；除此之外，只会改变满足 $j'' \in [1, n]$ 和 $j'' < j$ 的值 $v'_{(i-1)n+j''}$。然而，这种运行扩展不会改变总和 $\sum_{j=1}^{n} 2^{j-1} v'_{(i-1)n+j}$。也就是说，如果我们对每个 $j$ 和 $i$ 都迭代执行此操作，最终就会得到 $\mathbf{0}$。

### A.2 无界性谓词

这里我们讨论 [15] 中一维无界性谓词的概念与本文所用概念之间的关系。对于任意语言 $K,\ L \subseteq \Sigma^*$，[15] 中的公理集合如下：

(O1) 如果 $\mathfrak{p}(K)$ 且 $K \subseteq L$，则 $\mathfrak{p}(L)$。(O2) 如果 $\mathfrak{p}(K \cup L)$，则 $\mathfrak{p}(K)$ 或 $\mathfrak{p}(L)$。(O3) 如果 $\mathfrak{p}(F(K \cdot L))$，则 $\mathfrak{p}(F(K))$ 或 $\mathfrak{p}(F(L))$。

此外，在 [15]，中，所得的判定问题定义如下：

**输入** 语言 $L \subseteq \Sigma^*$。**问题** $\mathfrak{p}(F(L))$ 是否成立?

请注意，在此定义中，我们判定 $\mathfrak{p}(F(L))$ 是否成立；而第 2 节中的定义则是判定 $\mathfrak{p}(L)$ 是否成立。

1. 定义 (U1)–(U4) 稍强一些。注意，满足 (U1)–(U4) 的谓词 $\mathfrak{p}$ 也满足 (O1)–(O3)。不过，也存在一些（病态的）谓词示例，它们满足 (O1)–(O3)，但不满足 (U1)–(U4)。例如，谓词 $\mathfrak{p}_{\neg a}$，其

$$\mathfrak{p}_{\neg a}(L) \iff L \text{ is non-empty and } L \neq \{a\}$$

   满足 (O1)–(O3)，但不满足 (U4)。这是因为 $F(\{a\}) = \{\varepsilon, a\}$ 满足 $\mathfrak{p}_{\neg a}$，但 $\{a\}$ 本身并不满足。
2. 尽管 (U1)–(U4) 稍强一些，但它们导出的判定问题集合相同。事实上，假设 $\mathfrak{p}$ 是满足 (O1)–(O3) 的谓词。定义谓词 $\mathfrak{p}'$，使得 $\mathfrak{p}'(L) \iff \mathfrak{p}(F(L))$。那么，$\mathfrak{p}'$ 满足 (U1)–(U4)。此外，由 $\mathfrak{p}$ 和 $\mathfrak{p}'$ 导出的判定问题相同：两种情况下，我们都判定 $\mathfrak{p}(F(L))$ 是否成立。3. 因此，由公理集 (O1)–(O3) 和 (U1)–(U4) 导出的判定问题集合相同，所以定理 2 与所使用的公理集无关。

在 [15]中使用公理 (O1)–(O3) 的原因是，如果还引入多维无界性谓词，那么公理 (O1)–(O3) 与一般情形更为契合。[15] 中的定义有一个小缺点：必须判定 $\mathfrak{p}(F(L))$，而不是 $\mathfrak{p}(L)$，因此我们在这里选择了 (U1)–(U4)。

## B 第3节的补充材料

**将 $(k, r)$-RBCA 的转移转换为 $(k', 1)$-RBCA**。我们针对四种转移类型 $\mathrm{inc}_i$、$\mathrm{dec}_i$、$\mathrm{zero}_i$ 和 $\mathrm{nz}_i$ 给出了一个显式构造。

首先考虑转移 $(p, a, \text{递增}_i, q)$（图 1）。我们从一系列针对 $\mathrm{p}_{i,j}$ 的测试（以蓝色标记）开始，以确定当前所处的阶段。若对于奇数 $j$，$\mathrm{p}_{i,j}$ 非零，我们便知道当前处于正相位，只需递增相应的程序计数器 $\mathrm{c}_{i,\frac{j+1}{2}}$。否则，我们处于负相位。因此，在递增相应的程序计数器 $\mathrm{c}_{i,\frac{j}{2}+1}$之前，需要切换到正相位：递减 $\mathrm{p}_{i,j}$ 并递增 $\mathrm{p}_{i,j+1}$。

注意，在具有 $r$ 反转界的运行中，状态 $f$ 永远无法到达。由于相位计数器 $\mathrm{p}_{i,1}$ 到 $\mathrm{p}_{i,2m-1}$ 均已测试为负数，我们必定处于阶段 $r = 2m$。由于这是一个偶数阶段，计数器必定在递减，而递增意味着又增加一次反转，从而违反界限。

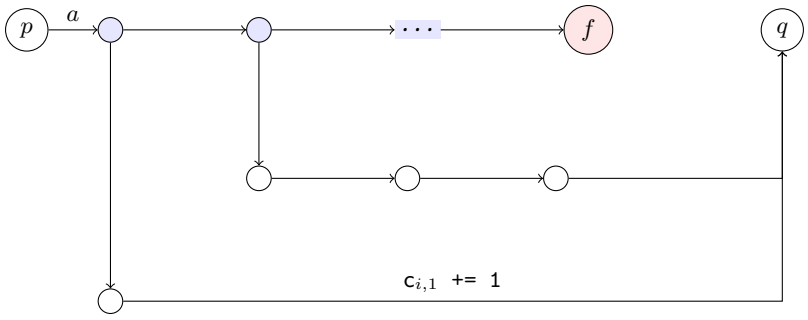


**图 1.** 用于替换递增转移的转移结构（$p$，$a$，$\mathrm{inc}_i$，$q$）。

接下来，考虑转移（$p$，$a$，$\mathrm{dec}_i$，$q$）（图 2）。与前面的构造类似，我们首先确定状态序列中以蓝色标记的当前阶段。这一次，如果当前处于正（奇数）阶段并转入负阶段，我们就改变阶段计数器。更新阶段计数器后，我们转到以绿色标记的状态，这些状态允许我们非确定性地递减某个程序计数器（回想一下，如果处于阶段 $j$，我们会选择计数器 $\mathrm{c}_{i,1}$，……，$\mathrm{c}_{i,\lceil \frac{j}{2} \rceil}$ 中的一个进行递减）。

最后，我们考虑零测试（图 3）和非零测试（图 4）。通过确保所有部分计数器 $\mathrm{c}_{i,j}$ 均为零，来测试计数器 $\mathrm{c}_i$ 是否为零；通过测试任一部分计数器 $\mathrm{c}_{i,j}$ 是否非零，来测试计数器 $\mathrm{c}_i$ 是否非零。

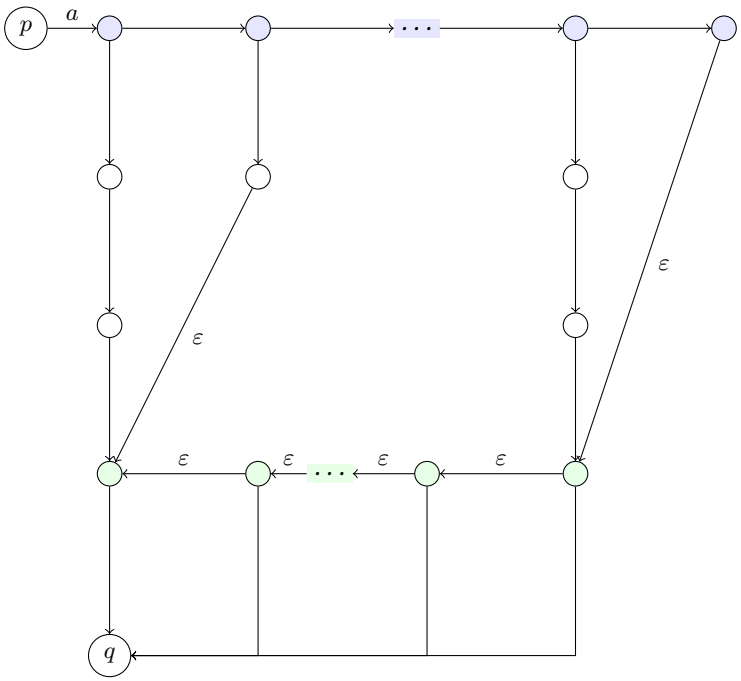


**图 2.** 用于替换递减转移的转移结构（$p$，$a$，递减$_i$，$q$）。

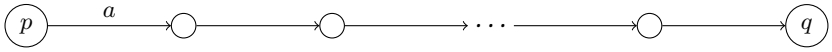


**图 3.** 用于替换零测试的转移结构（$p$，$a$，零$_i$，$q$）。

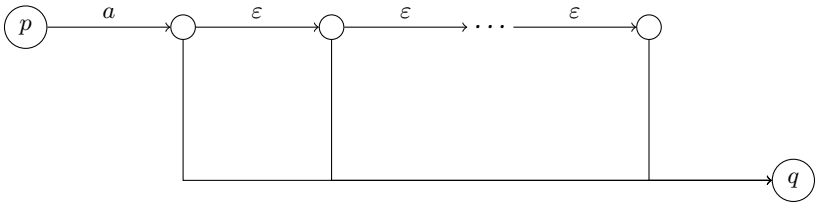


**图 4.** 用于替换非零测试的转移结构（$p$，$a$，$\mathrm{nz}_i$，$q$）。

## C 第4节的补充材料

首先注意，给定一个子集 $Q$，我们可以在对数空间内计算出一个上下文无关文法，用于表示 $L_{A,\ Q}$。为了计算这个语言，我们构造两个文法（一个用于 $A \overset{*}{\Rightarrow}_Q uAv$ 的 $u$ 部分，另一个用于 $A \overset{*}{\Rightarrow}_Q uAv$ 的 $v$ 部分）。然后，$L_{A,\ Q}$ 就是这两个文法所表示语言的并集。我们在此介绍 $u$-文法的构造，另一个可以对称地构造出来。

直观地说，对于 $Q$ 中的每个产生式，我们都将尝试猜测其生成的非终结符中，哪一个会生成带有悬空 $A$ 的子树。左侧的所有非终结符都将照常求值，右侧的所有内容（包括终结符）都将被丢弃，而选定的非终结符将继续猜测其子节点中的哪一个将负责生成 $A$。

形式化地说，我们按如下方式构造一个文法：

**1.** 将非终结符集合复制一份，$N := M \cup \{\hat{}X \mid X \in M\}$**2.** 保留所有原有产生式，并额外引入以下产生式：若 $X \to w$ 是来自 $Q$ 的产生式，且 $w$ 中至少有一个非终结符，那么对于沿某个非终结符拆分 $w$ 的所有方式 $uYv$，我们引入产生式 $X \to uY$。ˆ **3.** 引入产生式 $A \to \varepsilon$。ˆ **4.** 开始符号为 $A$。

ˆ 请注意，任何带标记的非终结符 $X$ 在任意（部分）推导中都只能出现在最右侧。它只能通过产生式 $A \to \varepsilon$被移除。因此，当且仅当在 $Q$中存在推导 $A \overset{*}{\Rightarrow}_Q uAv$ 时，这个新文法才能生成词 $u$。

我们还指出新文法结构的以下事实：

备注 *1.* 如果原文法的所有产生式都是左线性的（或都是右线性的），那么对于语言 $L_{A,Q}$，我们也能得到左线性（分别为右线性）文法。因此，我们可以为 $L_{A,Q}$ 构造非确定有限自动机，从而得到关于 $\mathbb{Z}$-向量加法系统的更优复杂度结果。

**引理 2.** 给定子集 $M \subseteq N$，我们可以在 NP 时间内检查 $M$ 是否可实现。此外，给定 $M \subseteq N$和 $p \in P$，我们可以在 NP 时间内检查 $p$ 是否可 $M$-消去。

证明。 我们首先证明，可以在 NP 时间内判定集合 $M \subseteq N$ 是否可实现。文献 [56], 证明了，可以在多项式时间内计算出一个存在型 Presburger 公式 $\psi$，使得向量 $\boldsymbol{u} \in \mathbb{N}^P$ 满足 $\psi$，当且仅当 $\boldsymbol{u}$对应于某个上下文无关文法的终结推导。我们通过加入约束来扩充该公式，这些约束要求对所有 $A \in M$，某个产生式 $A \to w$ 都会出现，且不会出现满足 $A \notin M$ 的产生式 $A \to w$；此外，还要求计数器总效应为零，然后检查该公式是否可满足。

为了检验产生式 $p$ 的 $M$-实数可实现性，我们首先构造第二个文法 $G'$。文法 $G'$ 模拟 $G$ 的泵。更准确地说，$G'$ 先模拟某个泵 $A_1 \overset{*}{\Rightarrow} u_1 A_1 v_1$，然后模拟某个泵 $A_2 \overset{*}{\Rightarrow} u_2 A_2 v_2$，依此类推，并在模拟有限个泵后停止。不过，它并不产生这些泵所生成的

终结符，而是将 $P$ 用作其终结符集，并在应用产生式 $p'$ 时产生字母 $p' \in P$。然后，由于 $P$ 是 $G'$ 的终结符字母表，对于每个 $\boldsymbol{u} \in \mathbb{N}^P$，我们都有：

$$\boldsymbol{u} \in \Psi(L(G')) \iff \exists k \in \mathbb{N},\ \text{pumps } \tau_1, \ldots, \tau_k \colon \boldsymbol{u} = \Psi(\tau_1) + \cdots + \Psi(\tau_k)$$

显然，可以在多项式时间内计算出这样的文法 $G'$。给定文法 $G'$，我们再次应用 [56] 的结果，计算一个存在型 Presburger 公式 $\psi$，该公式定义 $\Psi(L(G'))$。由此可知，$p$ 是 $M$-可实现的，当且仅当存在某个 $\boldsymbol{u} \in \Psi(L(G'))$，使得

1. $\boldsymbol{u}(p) > 0$，2. $\varphi(\boldsymbol{u}) = \boldsymbol{0}$，以及 3. $\boldsymbol{u}$ 中出现的所有产生式都属于 $M$。

由于借助 $\psi$，这样的 $\boldsymbol{u}$ 的存在性可以很容易地用另一个存在型 Presburger 公式 $\psi'$ 表示，因此只需检查 $\psi'$ 的可满足性，这可以在 NP 中完成。□

## D 第 5 节的补充材料

**关系 $\hookrightarrow_{\text{lin}}$ 和增长** 下面给出正文中用到的以下事实：

**引理 10。** 如果 $L_1 \hookrightarrow_{\text{lin}} L_2$ 和 $L_1$ 具有指数增长，那么 $L_2$ 也具有指数增长。

证明。我们首先断言，存在 $c > 0$，使得 $|L_1 \cap \Sigma^{\leq m}| \leq cm(m+1)\,|L_2 \cap \Sigma^{\leq cm}|$。由于 $L_1 \hookrightarrow_{\text{lin}} L_2$，存在一个 $c > 0$，使得对每个词 $w_1 \in L_1$，都存在一个词 $w_2 \in L_2$，满足 $|w_2| \leq c \cdot |w_1|$，并且 $w_1$ 是 $w_2$ 的因子。这使我们能够构造一个单射

$$f_m \colon L_1 \cap \Sigma^{\leq m} \to (L_2 \cap \Sigma^{\leq cm}) \times \{1, \ldots, cm\} \times \{0, \ldots, m\},$$

它将每个词 $w_1 \in L_1 \cap \Sigma^{\leq m}$ 映射为 $(w_2, r, |w_1|)$，其中 $w_2 \in L_2 \cap \Sigma^{\leq cm}$ 是由 $L_1 \hookrightarrow_{\text{lin}} L_2$ 得出的词，而 $r \in \{1, \ldots, cm\}$ 是 $w_2$ 中的一个位置，$w_1$ 在该位置出现。那么显然，$f_m$ 必定是单射。由于右侧集合的基数为 $cm(m+1) \cdot |L_2 \cap \Sigma^{\leq cm}|$，这就证明了该断言。

因此我们现在可以推断，$L_2$ 具有指数增长：由于 $L_1$ 具有指数增长，存在某个实数 $r > 1$，使得对于无限多个 $m$，都有 $|L_1 \cap \Sigma^{\leq m}| \geq r^m$。根据我们的断言，这意味着对于无限多个 $m$，都有 $|L_2 \cap \Sigma^{\leq cm}| \geq {}_{cm}{}^{r^m}\ {}_{(m+1)}$。由于对于几乎所有 $m$，都有 ${}_{cm}{}^{r^m}\ {}_{(m+1)} \geq (\sqrt[2c]{r})^{cm}$，我们特别可以得出，对于无限多个 $m$，都有 $|L_2 \cap \Sigma^{\leq cm}| \geq (\sqrt[2c]{r})^{cm}$。因此，$L_2$ 具有指数增长。□

**引理 9 的证明**

证明。 我们按照引理 3 的证明方法进行。不过，我们需要证明，每个词 $u$、$v$ 都可以作为某个词 $w \in L(G)$的因子，而该词的长度至多关于 $|u|$ 或 $|v|$呈线性增长（分别对应这两种情况）。我们对 $u$证明这一点； $v$ 的情形类似。首先，选取 $\ell \in \mathbb{N}$ ，使得对于每个非终结符 $B \in M$，都存在一个词$x \in \Sigma^*$ ，满足 $B \overset{*}{\Rightarrow}_Q x$ 和 $|x| \leq \ell$。由于 $A \overset{*}{\Rightarrow}_Q uAv$，我们还可以找到一个推导 $A \overset{*}{\Rightarrow}_Q uAv'$，其中 $|v'| \leq \ell \cdot |N| \cdot (|u|+1)$。这是因为在推导 $A \overset{*}{\Rightarrow}_Q uAv$中，从 $A$ 到 $A$-叶节点的路径上，至多有 $|u|$ 个左分支节点（即向左分支的节点）。通过删去泵，我们可以假设任意两个左分支节点之间（以及第一个左分支节点之上、最后一个左分支节点之下）至多有 $|N|$ 个节点。这样，我们便得到一个推导 $A \overset{*}{\Rightarrow}_Q uAv''$ 的推导树，其中至多有 $|N| \cdot (|u|+1)$个右分支节点。然后，将每个右分支节点下方的子树替换为一个推导长度为 $\leq \ell$ 的词的树，即可得到推导 $A \overset{*}{\Rightarrow}_Q uAv'$。

现在，我们将引理 3 的证明应用于推导 $A \overset{*}{\Rightarrow}_Q uAv'$。注意，在 $\tau_0$、 $\tau'_1$、.....、$\tau'_m$和 $\tau$中使用的产生式总数有界于$O(|u|+|v'|) = O(|u|)$：首先，注意 $\tau_0$ 和流 $\boldsymbol{f}_p$与 $u$和 $v'$相互独立。 $\boldsymbol{f}_\tau = \Psi(\tau'_1) + \cdots + \Psi(\tau'_m)$中的产生式总数相对于 $\tau$ 的产生式数量呈线性有界，这是由 (1) 得出的。若 $G$ 为乔姆斯基范式，则后者又有界于 $|u|+|v'|$ 。□

**多项式增长与指数增长的概念** 语言 $L \subseteq \Sigma^*$ 的多项式增长 [25], 定义为：存在一个多项式 $p(x)$，使得对于每个 $m \geq 0$，都有$|L \cap \Sigma^m| \leq p(m)$。此外，指数增长定义为存在一个实数 $r > 1$ ，使得对于无限多个$m \geq 0$，都有 $|L \cap \Sigma^m| \geq r^m$ 。这与我们的定义略有不同，我们的定义有时也称为累积增长 [12]。可以证明，事实上，就多项式增长和指数增长而言，这两种定义是等价的，参见例如 [24, 引理 2。3]下面我们给出证明，以求完整。

首先，如果存在一个多项式 $p(x)$，使得对所有 $m$ 都有 $|L \cap \Sigma^m| \leq p(m)$，那么我们也有$|L \cap \Sigma^{\leq m}| \leq \sum_{i=0}^{m} |L \cap \Sigma^i| \leq \sum_{i=0}^{m} p(i) \leq (m+1) \cdot p(m)$，这同样是一个多项式界。（注意，我们显然可以假设 $x \mapsto p(x)$ 是单调函数。）因此，根据我们的定义， $L$ 具有多项式增长。反向结论显而易见。

其次，假设存在一个实数 $r > 1$，使得对于无限多个 $m$，都有 $|L \cap \Sigma^{\leq m}| \geq r^m$。我们声称，此时对于无限多个 $m$，都有 $|L \cap \Sigma^m| \geq \sqrt{r}^m$。为导出矛盾，假设存在一个 $k \geq 0$ ，使得对于所有 $m \geq k$，都有$|L \cap \Sigma^m| < \sqrt{r}^m$。这意味着

$$r^m \leq |L \cap \Sigma^{\leq m}| = \sum_{i=0}^{m} |L \cap \Sigma^i| < M + \sum_{i=0}^{m} \sqrt{r}^i = M + \frac{\sqrt{r}^{m+1} - 1}{\sqrt{r} - 1} \qquad (2)$$

对于所有 $m \geq k$，其中 $M$ 是常数 $\sum_{i=0}^{k} |L \cap \Sigma^i| - \sqrt{r}^i$。最后一个相等关系由归纳法得出

$$\sqrt{r}^0 = 1 = \frac{\sqrt{r}-1}{\sqrt{r}-1} \quad \text{and} \quad \sum_{i=0}^{m+1} \sqrt{r}^i \overset{\text{IH}}{=} \frac{\sqrt{r}^{m+1}-1}{\sqrt{r}-1} + \sqrt{r}^{m+1} = \frac{\sqrt{r}^{m+2}-1}{\sqrt{r}-1}.$$

然而，当 $m$足够大时，显然不满足式（2）。反之亦然：如果 $|L \cap \Sigma^m| \geq r^m$，那么显然也有 $|L \cap \Sigma^{\leq m}| \geq r^m$。