arXiv:2603.08331v1 [cs.FL] 9 Mar 2026

# 上下文无关语言、下推自动机和单计数器自动机的转向复杂度*

Giovanni Pighizzini§米兰大学计算机科学系，意大利米兰
pighizzini@di.unimi.it

**摘要**

下推自动机的一次计算中的一个转向，是指下推存储器高度从增加阶段转为减少阶段。给定一个下推自动机或单计数器自动机，对于其语言中的每个字符串，我们考虑接受计算中的最小转向次数。我们证明，无法判定这个次数是否受某个常数所界。此外，我们还得到了如下自动机之间的非递归权衡：在有限次转向内接受的下推自动机和单计数器自动机，与有限转向下推自动机；后者的定义要求常数界对每个接受计算都成立。我们证明，存在一些语言，其接受所需的转向次数相对于输入长度是次线性的，但并非常数。此外，还存在一个无限且严格的复杂性类层级，其中转向次数分别受不同的次线性函数所界。另外，存在一种语言，其所需的转向次数并非常数，但增长速度慢于定义上述层级的每个函数。

*本文的初稿版本曾发表于 DLT 会议论文集 2025 [Pig25].§作者是国家科学计算组（Gruppo Nazionale Calcolo Scientifico）—国家高等数学研究所（Istituto Nazionale di Alta Matematica，GNCS-INdAM）的成员。

# 1 引言

最近，我们研究了与上下文无关语言、下推自动机和单计数器自动机有关的两种复杂性度量。下推存储器高度 度量下推存储器 [PP23] 所使用的空间。在单计数器自动机这一受限情形中，下推存储器仅用于表示以一元形式书写的计数器，这一度量对应于计数器达到的最大值 [PP25]。压入复杂度 则考虑计算中所用压入操作的次数 [Pig26]。显然，当其中一种度量受常数所界，也就是不依赖于输入长度时，下推存储器可能具有的配置数目有限。因此，可以将存储器编码到有限控制中。这意味着所接受的语言具有正则性。

非常数情形更有意思。首先，若采用所谓的强度量，将所有计算的开销都纳入考量；或者对每个被接受的输入，仅考虑开销较低的接受计算（弱度量），两者之间存在根本差异。在强度量下，如果上述资源中的任一种不受任何常数限制，那么就需要使用随输入长度线性增长的资源量 [PP25, Pig26]。如果考量所有接受计算（而非所有计算）的开销，也有类似结果（接受度量） [PP25]。

弱度量的情况更复杂，但也更具挑战性。我们指出，这种度量与广义上的非确定性概念密切相关：非确定型机器不仅能够在存在接受策略时作出最终导致接受的选择，也能够作出最终导向开销较低的接受策略的选择。对于开篇提到的两种资源，已有证明表明，如果机器使用的资源量不受任何常数限制，那么该资源量至少应随输入长度以双重对数函数的速度增长。对于包含至少两个符号的输入字母表，这一下界是最优的；而在一元情形下，则得到了不同且更高的下界（具体介于对数级和线性级之间，取决于资源类型）。

本文通过考察计算中进行的转

向次数这一额外资源，继续沿着这条研究思路展开。粗略地说，在一次计算中，当存储栈高度增加的一系列移动之后，接着出现高度降低的一系列移动时，下推自动机便完成了一次转向。在一篇关于形式语言和下推自动机的开创性论文中，Ginsburg 和 Spanier 开始研究转向在下推自动机计算中的作用 [GS66]。特别是，他们引入了有限转向下推自动机，即存在一个常数来限制每次接受计算中转向次数的下推自动机。这些装置所接受的语言类是真正包含于上下文无关语言中的真子类，并且真包含正则语言类。即使将范围限制为只进行一次转向的单计数器自动机，所得语言类也比正则语言类更广。特别是，一转次下推自动机刻画了线性上下文无关语言类。

这些事实已经揭示了转向次数与下推高度、压入复杂度等资源之间的一个主要区别：只需一次转向，其能力就已经超越了仅有有限控制器的情形。在同一篇论文中，Ginsburg 和 Spanier 给出了一些有趣的结果。其中，他们证明，对于给定的 $k$，判断一个下推自动机是否为有限转向，以及是否为 $k$-转向，是可判定的。他们还通过一种称为超线性文法的特殊上下文无关文法，对有限转向下推自动机接受的语言进行了刻画。因此，有限转向自动机接受的语言类称为超线性语言类。需要指出的是，这里考虑机器所有接受计算，也就是说，采用的是接受度量。

很自然会想到，考虑弱度量时会发生什么；也就是说，对于每个被接受的输入，仅考虑使用最少转向次数的接受计算。下文中，当我们说一台机器以特定转向次数接受时，始终指弱度量。Ginsburg 和 Rice 观察到，如果一台机器以常数次转向接受，即对于每个被接受的输入，都存在一个包含至多 $k$ 次转向的接受计算，那么可以修改该机器，使其在转向次数超过 $k$ 时停止。由此得到的$k$转向下推自动机与原机器等价。因此，即使在弱度量下，以有限次转向接受的语言类也与超线性语言类相同。然而，与

能够判定下推自动机是否为有限转向（接受度量）这一事实相反，Ginsburg 和 Rice 还证明了，无法判定下推自动机是否接受超线性语言 [GS66]。

在进一步深入这一研究时，于本文第一部分，我们在第2节给出预备概念以及一条将在文中若干证明中发挥关键作用的引理之后，在第3节证明：一个下推自动机是否以有限次转向接受（弱度量）是不可判定的。[1] 实际上，我们证明该性质即使在单计数器自动机这一受限情形下也已不可判定。我们还证明，对于给定的 $k > 0$，无法判定一个单计数器自动机是否以 $k$轮次接受。对这些装置以及下推自动机而言，只有以0轮次接受才是可判定的。

我们曾提到，每个以 $k$ 轮次接受的下推自动机都可以加以改造，从而得到一个等价的 $k$转向下推自动机。然而，正如我们在文中所证明的，值 $k$ 不能被关于给定机器规模的任何递归函数所界定。因此，我们能够证明，以常数转向次数接受的下推自动机与有限转向下推自动机之间的规模权衡是非递归的。[2] [3]

本文的第二部分致力于研究在仍采用弱度量的情况下，转向次数如何随输入长度增长。很容易给出在常数转向次数下接受的语言，以及在转向次数呈线性数量增长的情况下接受的语言。这里，我们证明了对于若干个次线性函数，存在在次线性转向次数下接受的语言。首先，在第 4 节中，我们证明存在一种语言，单计数器自动机可在 $O(\sqrt{n})$ 次转向下接受该语言，其中 $n$ 是输入的长度。此外，即使使用下推自动机，该语言也不能在 $o(\sqrt[3]{n})$ 次转向下被接受。

—

我们随后证明，存在使用甚至更低的机器和语言

[1]我们指出，这个问题不同于上述判定下推自动机是否接受超线性语言的问题。事实上，我们可以构造接受超线性语言，甚至接受正则语言的下推自动机，但其使用的转向次数是无界的。[2]关于不可递归权衡的讨论，读者可参阅 [Kut05]。[3]关于转向次数在下推自动机计算中的作用，在非确定性和确定性情形下，Malcher [Mal07]均已取得进一步结果，相关研究并在 [MP13] 中得到拓展。

转向次数。在第 5 节中，我们给出一种构造：给定一个以 $t(n)$ 轮次接受某一语言的下推（相应地，单计数器）自动机，便可得到一个以 $t$(对数 $n$) 轮次接受另一种语言的下推（相应地，单计数器）自动机。通过迭代这一构造，我们可以任意减少转向次数。特别地，通过对该构造加以改进，对于每个整数 $k$ ，我们证明存在语言 $\mathrm{L}_k$ ，它可由单计数器自动机以 $\lg^{(k)} n$ 轮次接受，但不能由任何下推自动机以增长慢于 $\lg^{(k)} n$ 的转向次数接受。[4] 这一结果揭示了它与下推高度和压入复杂度之间的重大差异——当后两者不为常数时存在双对数下界——并相对于转向次数定义了复杂性类的一个严格无限层次。

还可以进一步降低：在第 6 节中，我们证明存在一种语言 $\mathrm{U}_*$ ，它可由单计数器自动机在 $\lg^* n$轮次内接受（ $n$ 的迭代对数 ，这是一种增长速度慢于 $\lg^{(k)} n$ 的函数，对任意 $k$ 均成立）。我们还证明，这样的轮次数是必要的。

最后，我们在第7节中给出一些总结性评论。

## 2 预备知识

我们假设读者熟悉教科书中介绍的自动机和形式语言理论的标准概念，例如 [HU79]。给定一个字母表 $\Sigma$，其上的字符串集合记为 $\Sigma^*$，其中空字符串记为 $\varepsilon$。字符串 $x \in \Sigma^*$ 的长度记为 $|x|$，而符号 $a \in \Sigma$ 在 $x$ 中出现的次数记为 $|x|_a$。给定集合 $S$，其基数记为 $\#S$，其子集族记为 $2^S$，其补集记为 $S^c$。

令$\log_2 n$ 表示以2为底的对数。出于技术原因，我们修改该函数，以使其具有非负 值域。更确切地说，我们考虑lg函数 $n$，其对 $n \geq 0$ 的定义如下：

$$\lg n = \begin{cases} \log_2 n & \text{if } n \geq 1, \\ 0 & \text{otherwise.} \end{cases}$$

给定一个整数 $k \geq 0$，$\lg^{(k)}$表示通过复合 $k$ 得到的函数

[4] $\lg^{(k)}$表示以2为底的对数与自身的 $k$ 次复合。

次 lg函数与其自身，即 $\lg^{(k)} n = \lg \lg \cdots \lg n$。我们指出$\underbrace{k \text{ 次}}$ $\lg^{(0)}$是恒等函数。

迭代对数是函数 $\lg^*$: $\mathbb{N} \to \mathbb{N}$ 定义为

$$\lg^* n = \min\{k \mid \lg^{(k)} n \leq 1\}.$$

该函数增长得非常缓慢。例如，对于 $16 < n \leq 65536$，$\lg^* n = 4$；而对于 $65536 < n \leq 2^{65536}$，$\lg^* n = 5$。

给定整数 $x, k \geq 0$，我们将数 $x^{x^{\cdot^{\cdot^{\cdot^{x}}}}}$（$\underbrace{k \text{ 次}}$）记为 $x^{\uparrow\uparrow k}$，即，

$$x^{\uparrow\uparrow k} = \begin{cases} 1 & \text{if } k = 0, \\ x^{(x^{\uparrow\uparrow k-1})} & \text{otherwise.} \end{cases}$$

不难观察到，对于 $0 \leq j \leq k$，$\lg^{(j)}(2^{\uparrow\uparrow k}) = 2^{\uparrow\uparrow k-j}$，因此 $\lg^{(k)}(2^{\uparrow\uparrow k}) = 1$。

本文将使用以下不等式:

**引理 1** 设 $\alpha > \beta > 1$，则 $\lg(\alpha\beta) < 2$对数 $\alpha$，并且对于 $k \geq 2$:

$$\lg^{(k)}(\alpha\beta) < 1 + \lg^{(k)}\alpha.$$

证明。 首先我们观察到 $\lg(\alpha\beta) < \lg \alpha^2 = 2\lg \alpha$。

对于 $k = 2$，我们有:

$$\lg^{(2)}(\alpha\beta) = \lg\lg(\alpha\beta) < \lg\lg\alpha^2 = \lg(2\lg\alpha) = 1 + \lg^{(2)}\alpha.$$

然后我们考虑 $k > 2$。通过归纳，我们得到:

$$\lg^{(k)}(\alpha\beta) = \log\lg^{(k-1)}(\alpha\beta) < \log(1 + \lg^{(k-1)}\alpha) \leq 1 + \lg^{(k)}\alpha,$$

其中，在最后一步，我们使用了如下事实：对于 $x \geq 1$，$\lg(1+x) \leq 1 + \lg x$。

□

我们简要回顾 [GS66, HU79] 中介绍的下推自动机概念。下推自动机（简称 PDA）定义为 $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0)$，其中 $Q$、$\Sigma$ 和 $\Gamma$ 是有限集，分别为状态集、输入字母表和下推字母表； $q_0 \in Q$ 是初始

状态； $Z_0 \in \Gamma$ 是下推存储器上的起始符号； $\delta$ 是从 $Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma$ 到 $Q \times \Gamma^*$ 的有限子集的转移函数，其中 $(p, \alpha) \in \delta(q, \sigma, A)$，若 $p, q \in Q$、$\sigma \in \Sigma \cup \{\varepsilon\}$、$A \in \Gamma$、$\alpha \in \Gamma^*$，则表示自动机处于状态 $q$，下推存储器栈顶为 $A$，并从纸带读取符号 $\sigma \in \Sigma$（或者在 $\sigma = \varepsilon$ 时不读取任何符号）后，可以转移到状态 $p$，同时将栈顶的 $A$ 替换为字符串 $\alpha$。[5] 我们假设采用空栈接受，也就是说，只有存在一条从初始配置（输入头位于 $w$ 的第一个符号上，有限控制处于状态 $q_0$，下推存储器中仅含有 $Z_0$）开始的计算，按照转移函数到达某个配置，且此时输入已全部扫描完毕（即输入头位于 $w$ 最右侧符号的右边）、下推存储器为空，输入 $w$ 才被接受。众所周知，这些装置所接受的语言类与上下文无关语言类相同。

上下文无关语言的计算中的转向 $\mathcal{C}$ 是由一系列 $k \geq 2$ 移动组成的，其中下推存储器的高度在第一次移动时增加，在最后一次移动时减少，而在中间保持不变。也就是说，转向由 $k+1$ 个连续配置定义，其中下推存储器的内容依次为 $\gamma_0$，$\gamma_1$，……，$\gamma_k$和 $|\gamma_0| < |\gamma_1| = |\gamma_2| = \cdots = |\gamma_{k-1}| > |\gamma_k|$。

根据 [GS66], ，当整数 $k \geq 0$满足的每个接受计算 至多包含 $k$ 次转向时，PDA $\mathcal{M}$ 称为 $k$-转向。若 PDA $\mathcal{M}$对于某个 $k \geq 0$是 $k$-转向的，则称其为有限转向。容易看出，0 转向的PDA仅识别正则语言。此外，1 转向 PDA所接受的语言类就是线性语言。这一语言类严格包含于有限转向 PDA所接受的语言类；后者就是超线性语言，是上下文无关语言的真子类。

在上述定义中，考虑了每次接受计算的转向次数。由于非确定型机器可能存在采用不同转向次数的多种计算，我们将引入另一种概念。具体而言，我们将考虑足以接受输入字符串所需的转向次数。更准确地说，我们给出如下定义：

**定义 1** 设 $\mathcal{M}$ 是一个接受语言 $L$的 PDA 。我们称一个

[5]我们约定，下推字符串按从上到下的顺序书写；因此，当 $\alpha \neq \varepsilon$ 时， $\alpha$ 的最左符号将在这次转移后位于栈顶。

字符串 $w \in L$ 在 $k$ 轮次内被接受，对于整数 $k \geq 0$，当存在 $\mathcal{M}$ 的一个计算，该计算接受 $w$ 且至多包含 $k$ 轮次。

给定一个函数 $t : \mathbb{N} \to \mathbb{N}$，我们说 $\mathcal{M}$接受 $L$ 以 $t(n)$ 轮次，是指对于每个满足 $|w| = n$ 的 $w \in L$，在 $w$ 上都存在一个至多包含 $t(n)$ 轮次的接受计算。此外，若函数 $t(n)$ 以常数为界，则我们说 $\mathcal{M}$ 以有限次转向接受。

如果我们知道某个 PDA 对某个给定的 $k \geq 0$ 以 $k$ 轮次接受，那么我们总可以在有限控制中对转向次数进行计数，并在该次数超过 $k$时停止计算，从而得到一个等价的 $k$ 转向 PDA 。然而，正如我们将看到的，除非 $k = 0$，否则以 $k$ 轮次接受是不可判定的。

根据 [Har78], ，一个 PDA 的大小是用于写下其描述的符号数目，并且它取决于其字母表和状态集的基数，以及出现在转移规则右部的下推符号的最大数目。

单计数器自动机（简称 OCA）是一种下推自动机，其下推字母表除 $Z_0$ 外仅由一个符号 $A$ 组成。符号 $Z_0$ 仅用于标记底部。这样，下推存储器始终包含形如 $A^k Z_0$ 的字符串，表示整数 $k \geq 0$，但接受配置除外，此时存储器为空。众所周知，由 OCA 所识别的语言类是上下文无关语言的真子类，并且真包含正则语言类。

在本文中，我们使用语言 $\mathbf{Eq} = \{a^n b^n \mid n > 0\}$，它是一种线性上下文无关语言，由一个平凡的确定性 OCA $\mathcal{A}_{\mathsf{Eq}}$ 接受，该自动机在每次接受计算中恰好进行一次转向。我们用 $\mathcal{A}_{\mathsf{Eq}^*}$ 表示接受 $\mathbf{Eq}$ 的克莱尼闭包 $\mathbf{Eq}^*$ 的平凡确定性 OCA ，并且它恰好进行 $k$ 轮次来接受 $\mathbf{Eq}^k$ 中的每个字符串，$k > 0$。众所周知的结论是，对任意固定的 $k \geq 0$， $k$ 轮次对于接受 $\mathbf{Eq}^k$ 是必要且充分的。此外，每个接受 $\mathbf{Eq}^*$ 的 PDA 都应当进行 $k$ 轮次，以接受任何形如 $a^{n_1} b^{n_1} a^{n_2} b^{n_2} \cdots a^{n_k} b^{n_k}$ 的字符串，其中 $n_1, n_2, \ldots, n_k$ 充分大。

在引理 3 中，我们提出了这些下界的一个推广，该推广将在本文中多次使用。

为了证明这一点，将每个 PDA 转换为一种范式会更方便：在这种范式中，下推存储器的最顶端符号不能在一次移动中被

替换。可以通过以下方式轻松得到这种范式：将每次把最顶端符号 $A$ 替换为字符串 $\alpha = \alpha' B$, $B \neq A$ 的移动，模拟为一次弹出 $A$ 的移动，随后再进行一次压入字符串 $\alpha'$ 的移动。然而，这种构造无法保持转向次数。在下面的技术引理中，我们将给出一种能够保持转向次数的替代模拟：

引理 2 对于每个 PDA $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0)$，都存在一个等价的 PDA $\mathcal{M}' = (Q', \Sigma, \Gamma', \delta', q_0', \bot)$，使得如果 $(p, \alpha B) \in \delta'(q, \sigma, A)$，其中 $p, q \in Q'$，$\alpha \in \Gamma'^*$，$\sigma \in \Sigma \cup \{\varepsilon\}$，$A, B \in \Gamma'$，则 $A = B$。此外，对于每个 $w \in \Sigma^*$ 和 $k \in \mathbb{N}$，$\mathcal{M}$ 在 $w$ 上恰好进行 $k$ 轮次的接受计算，当且仅当 $\mathcal{M}'$ 在 $w$ 上恰好进行 $k$ 轮次的接受计算。

证明。 PDA $\mathcal{M}'$ 通过在其有限控制器中保存给定PDA $\mathcal{M}$的下推存储器最顶端的符号，直接模拟其移动。因此，$\mathcal{M}$ 的下推内容由有限控制器中的符号以及 $\mathcal{M}'$存储器中的符号表示。为此，当 $\mathcal{M}$ 的一次转移将最顶端符号 $A$ 替换为非空字符串 $\alpha$时，在 PDA $\mathcal{M}'$ 中，保存在控制器中的符号 $A$被替换为 $\alpha$的第一个符号，而 $\alpha$ 中的其他符号则被压入栈中。当 $\mathcal{M}$ 的一次转移将栈中的最顶端符号弹出时，$\mathcal{M}'$ 中相应的转移也会弹出最顶端符号；该符号是 $\mathcal{M}$的“新”顶端，并将其保存到控制器中。出于技术原因，引入一个新的下推符号 $\bot$很有用，该符号用于标记 $\mathcal{M}'$存储器的底部。只有当 $\mathcal{M}$ 存储器变空时，才会移除该符号，从而结束计算，因为下一次转移未定义。

形式上，我们取 $\Gamma' = \Gamma \cup \{\bot\}$，其中 $\bot \notin \Gamma$ 是计算开始时下推存储器上的唯一符号，且 $Q' = Q \times \Gamma'$。为简洁起见，$Q'$ 中的状态表示为 $[qA]$，其中 $q \in Q$ 且 $A \in \Gamma'$。初始状态 $q_0'$ 为 $[q_0 Z_0]$。

转移函数 $\delta'$： $Q' \times (\Sigma \cup \{\varepsilon\}) \times \Gamma' \to 2^{Q' \times \Gamma'^*}$ 通过考虑下面列出的转移来定义，其中 $\sigma \in \Sigma \cup \{\varepsilon\}$、$q, p \in Q$、$A \in \Gamma$、$\alpha \in \Gamma^*$:

(a) 对于每个转移 $(p, \varepsilon) \in \delta(q, \sigma, A)$，以及对于每个 $C \in \Gamma \cup \{\bot\}$,$\delta'([qA], \sigma, C)$，其中包含 $([pC], \varepsilon)$;

(b) 对于每个转移 $(p, B_1B_2\cdots B_k) \in \delta(q, \sigma, A)$，其中 $B_1, B_2, \ldots, B_k \in \Gamma, k > 0$，且对于每个 $C \in \Gamma \cup \{\perp\}$, $\delta'([qA], \sigma, C)$ 包含 $([pB_1], B_2\cdots B_kC)$。

现在我们采用 PDAs 的配置和移动的标准记法，如[HU79]中所示：配置是一个三元组 $(q, x, \alpha)$，其中 $q$是当前状态，$x$是尚未读取的输入部分，也就是说，是从当前输入头位置开始的部分；$\alpha$ 是下推存储器上的字符串，其中最顶端符号位于开头；而 $(q, x, \alpha)$ $\mathcal{N}(q', x', \alpha')$ 表示从配置 $(q, x, \alpha)$ 出发，一步之内 PDA $\mathcal{N}$可以到达配置 $(q', x', \alpha')$。关系 $\mathcal{N}$ 的自反传递闭包记作 $\overset{*}{\underset{\mathcal{N}}{\vdash}}$。

由 (a) 可知，对于 $q, p \in Q$, $\sigma \in \Sigma \cup \{\varepsilon\}$, $x \in \Sigma^*$,$A, C_1, C_2, \ldots, C_s \in \Gamma$，若 $s > 0$，则：

$$(q, \sigma x, AC_1C_2\cdots C_s) \vdash_{\mathcal{M}} (p, x, C_1C_2\cdots C_s) \text{ if and only if}$$
$$([qA], \sigma x, C_1C_2\cdots C_s\perp) \vdash_{\mathcal{M}'} ([pC_1], x, C_2\cdots C_s\perp),$$

而当 $s = 0$ 时：

$$(q, \sigma x, A) \vdash_{\mathcal{M}} (p, x, \varepsilon) \text{ if and only if } ([qA], \sigma x, \perp) \vdash_{\mathcal{M}'} ([p\perp], x, \varepsilon).$$

由 (b) 可得，对于 $q, p \in Q, \sigma \in \Sigma \cup \{\varepsilon\}, x \in \Sigma^*, A, B_1, B_2, \ldots, B_k, C_1, C_2, \ldots, C_s \in \Gamma$，其中 $k > 0, s \geq 0$:

$$(q, \sigma x, AC_1C_2\cdots C_s) \vdash_{\mathcal{M}} (p, x, B_1B_2\cdots B_kC_1C_2\cdots C_s) \text{ if and only if}$$
$$([qA], \sigma x, C_1C_2\cdots C_s\perp) \vdash_{\mathcal{M}'} ([pB_1], x, B_2\cdots B_kC_1C_2\cdots C_s\perp).$$

因此，对于 $\mathcal{M}$的每个连续格局序列:

$$(q_1, w_1, A_1\alpha_1) \vdash_{\mathcal{M}} (q_2, w_2, A_2\alpha_2) \vdash_{\mathcal{M}} \cdots \vdash_{\mathcal{M}} (q_k, w_k, A_k\alpha_k)$$

都对应着 $\mathcal{M}'$的配置序列:

$$([q_1A_1], w_1, \alpha_1\perp) \vdash_{\mathcal{M}} ([q_2A_2], w_2, \alpha_2\perp) \vdash_{\mathcal{M}} \cdots \vdash_{\mathcal{M}} ([q_kA_k], w_k, A_k\perp)$$

反之亦然。注意，对于 $i = 1, \ldots, k$, $|A_i\alpha_i| = |\alpha_i\perp|$。这意味着两个序列包含完全相同的转向次数。此外，如果将第一个序列中的最终格局替换为 $(q_k, w_k, \varepsilon)$，也就是说，最后一次移动将 $A_{k-1}$ 从下推自动机中弹出，并且 $\alpha_{k-1} = \varepsilon$，那么相应序列中的最终格局就变为 $([q_k\perp], w_k, \varepsilon)$。

由此可得，对于每个 $w \in \Sigma^*$，$(q_0, w, Z_0) \overset{*}{\vdash}_{\mathcal{M}} (q, \varepsilon, \varepsilon)$ 当且仅当 $([q_0 Z_0], w, \bot) \overset{*}{\vdash}_{\mathcal{M}'} ([q\bot], \varepsilon, \varepsilon)$，也就是说，$\mathcal{M}$ 和 $\mathcal{M}'$ 接受的语言相同。根据前面的讨论，我们还可以得出：$\mathcal{M}$ 有一项计算，恰好经过 $t$ 次转向接受字符串 $w$，当且仅当 $\mathcal{M}'$ 有一项计算，恰好经过 $t$ 次转向接受 $w$。

□

我们现在能够证明，接受语言 $\mathsf{Eq}^k$、$k > 0$ 的某些推广所需转向次数的一个下界。

**引理 3** 设 $\Sigma = \Delta \cup \{a, b\}$ 为一个字母表，其中含有 $a$、$b \notin \Delta$，并设 $\mathcal{M}$ 为一个接受语言 $L \subseteq \Sigma^*$ 的PDA。存在一个整数 $N > 0$，使得对每个整数 $k > 0$ 以及字符串 $y_0$、$y_1$、……、$y_k \in \Delta^*$，若语言 $L' = y_0\{a, b\}^* y_1 \{a, b\}^* \cdots y_{k-1}\{a, b\}^* y_k \cap L$ 是 $y_0\mathsf{Eq}^* y_1 \mathsf{Eq}^* \cdots y_{k-1}\mathsf{Eq}^* y_k$的子集，则对 $n_1$、$n_2$、……、$n_k \geq N' = N \cdot (|y_0 y_1 \cdots y_k| + 1)$，$\mathcal{M}$ 在输入 $y_0 a^{n_1} b^{n_1} y_1 a^{n_2} b^{n_2} \cdots y_{k-1} a^{n_k} b^{n_k} y_k$上的每个接受计算至少包含 $k$轮次。[6] [7]

证明。不失一般性，根据引理 2，我们可以假设，对于 $\mathcal{M}$ 的每个形如 $(q, \alpha) \in \delta(q, \sigma, A)$ 的转移，其中 $\alpha \neq \varepsilon$，$\alpha$ 的最后一个符号是 $A$。

从 $\mathcal{M}$出发，通过应用与正则语言求交的标准构造，我们可以得到一个 PDA $\mathcal{M}'$ 接受 $L'$，使得对于属于 $L'$的输入，$\mathcal{M}$与 $\mathcal{M}'$ 的接受计算之间存在一一对应关系。$\mathcal{M}$中的一个接受计算与 $\mathcal{M}'$ 中对应的计算恰好具有相同的转向次数。

我们补充说明一下证明中会用到的 $\mathcal{M}'$。取一个接受语言 $R = y_0\{a, b\}^* y_1 \{a, b\}^* \cdots y_{k-1}\{a, b\}^* y_k$ 的有限自动机 $A$。$A$ 的转移图由一条包含 $|y_0 y_1 \cdots y_k| + 1$ 个状态的路径构成，从初始状态到最终状态。该路径上各弧所标记字母的连接结果为 $y_0 y_1 \cdots y_k$。图中唯一的环是自环，分别标记为字母 $a$ 和 $b$，它们位于对应于

[6] 请注意，该陈述并不意味着在 $y_0 a^{n_1} b^{n_1} y_1 a^{n_2} b^{n_2} \cdots y_{k-1} a^{n_k} b^{n_k} y_k$上存在接受计算。[7] 我们在定理3、5、8和11的证明中使用这个引理。实际上，我们得到的语言 $L' = y_0\{a, b\}^* y_1 \{a, b\}^* \cdots y_{k-1}\{a, b\}^* y_k \cap L$ 总是 $y_0\mathsf{Eq}y_1\mathsf{Eq}\cdots y_{k-1}\mathsf{Eq}y_k$的一个子集。不过，我们更倾向于保留这一更一般的表述，因为它是由证明推导而来的。

因子 $y_{i-1}$ 末尾和 $y_i$ 开头的状态上，其中 $i = 1, \ldots\ldots, k$。因此， $\mathcal{M}'$ 的状态是形如 $(q, r)$ 的有序对，其中 $q$ 是 $\mathcal{M}$ 的一个状态，而 $r$ 是 $A$ 的一个状态。令 $N = (\#Q)^2 \cdot \#\Gamma$，其中 $Q$ 和 $\Gamma$ 分别是 $\mathcal{M}$ 的状态集和工作字母表。

假设 $x = y_0 a^{n_1} b^{n_1} y_1 a^{n_2} b^{n_2} \cdots y_{k-1} a^{n_k} b^{n_k} y_k$，其中 $n_1, n_2, \ldots\ldots, n_k \geq N' = N \cdot (|y_0 y_1 \cdots y_k| + 1)$，由 $\mathcal{M}$ 在少于 $k$ 轮次的情况下接受。那么也存在一个计算 $\mathcal{C}$，使 $\mathcal{M}'$ 在少于 $k$ 轮次的情况下接受 $x$。我们将证明这会导致矛盾。

由于 $\mathcal{C}$ 中的转向次数少于 $k$，必定存在一个指标 $i, 1 \leq i \leq k$，使得在消耗因子 $a^{n_i} b^{n_i}$时不进行任何转向。因此，消耗 $a^{n_i} b^{n_i}$ 的 $\mathcal{C}$ 部分 $\mathcal{C}'$ 不可能在执行压入操作之后包含任何弹出操作。这引出了两种情况。

如果在 $\mathcal{C}'$ 中，输入头到达因子 $b^{n_i}$之前执行了压入操作，那么在读取该因子时，下推存储器的高度不会降低。对于 $j = 0, \ldots, n_i$，考虑整数 $h_j$ 和五元组 $[q_j r_j A_j p_j s_j]$：其中，对于 $j = 0, \ldots, n_i$， $h_j$ 表示在 $\mathcal{C}'$中读取前缀 $a^{n_i} b^j$ 后，下推存储器的高度；$(q_j, r_j)$是 $\mathcal{M}'$控制器的状态； $A_j$ 是存储器顶端的符号；而$(p_j, s_j)$是 $\mathcal{C}$ 中消耗完因子 $b^{n_i}$ 之后的第一个配置中的状态，此时下推存储器的高度为 $h_j$。（大致来说，在读取 $b^{n_i}$时，下推存储器的高度可能会超过 $h_j$，因此在计算的下一部分中，经过若干次弹出操作后，下推存储器的高度应当再次降至 $h_j$。）我们注意到，由于上述定义的五元组中，前两个分量对应于同一因子 $b^{n_i}$上的计算部分，因此， $A$ 在读取 $b^j$ 之后的状态所对应的分量 $r_j$不会改变。因此，对于给定的 $b^{n_i}$，我们所考虑的不同五元组的数量有上界：属于 $(\#Q)^2 \cdot \#\Gamma$ 的三元组数量乘以分量 $s_j$可能取值的数量，即 $|y_0 y_1 \cdots y_k| + 1$。由于 $n_i \geq N'$，我们所考虑的五元组数量超过了这个上界。因此，存在两个指标 $j, \ell$，满足 $0 \leq j < \ell \leq n_i$，使得 $[q_j r_j A_j p_j s_j] = [q_\ell r_\ell A_\ell p_\ell s_\ell]$。此外， $h_j \leq h_\ell$。

如果 $h_j < h_\ell$ 那么这就定义了一个“垂直循环” [PP23], ，它在递增阶段消耗一个输入因子 $b^{\ell - j}$，并在递减阶段消耗一个输入因子 $z$。这个循环可用于泵送计算 $\mathcal{C}$，从而得到一个接受字符串 $x' \notin L'$ 的计算 $\mathcal{M}'$，这就构成了

矛盾。

否则 $h_j = h_\ell$，这就意味着非空因子 $b^{\ell-j}$ 在状态 $q_j$ 两次重复之间被消耗，且期间下推存储器保持不变。这样也可以通过将这两次重复之间的部分加倍来泵送计算 $\mathcal{C}$，从而再次得到字符串 $x' \notin L'$ 上的接受计算，矛盾。

如果在 $\mathcal{C}'$中，输入头到达因子 $b^{n_i}$之前 没有执行任何压入操作，那么在读取$a^{n_i}$时，栈的高度只能下降或保持不变。采用与前述论证对称的方法，我们便可得到矛盾。

□

## 3 不可判定性与非递归权衡

Ginsburg 和 Spanier 证明，对于任意给定的整数 $k \geq 0$ [GS66]，可以判定一个 PDA是否为有限转向，或是否为 $k$-转向。需要强调的是，计算转向次数时要考虑所有接受计算。因此，对于每个被接受的输入，都必须满足这一界限，即使是代价最高的接受计算也不例外。

在此，我们考虑足以接受给定输入的转向次数。因此，对于每个被接受的输入，该界限必须由使用最低转向次数的接受计算来满足。采用这一度量，我们得到完全不同的结果。特别地，除0次转向的接受外，上述所有问题都变得不可判定。我们还将证明一个非递归权衡结果，即表明：在接受所需的转向次数由某个值 $k$界定、且该值相对于输入长度而言为常数的情形下，该值不能由机器描述规模上的任何递归函数来界定。这将使我们能够证明，在有限次转向内接受的 OCA与 PDA以及有限转向 PDA之间的权衡不是递归的。

让我们首先介绍不可判定性结果，这些结果是通过改编Hartmanis提出的一种著名技术而得到的，该技术基于对单带图灵机计算的编码 [Har67]。

单带图灵机 $\mathcal{T}$的配置，其中状态集为 $Q$，带字母表为 $\Gamma$，按照标准方式表示为 $\Gamma^*Q\Gamma^*$中的字符串。一次计算编码为字符串 $\alpha_1\$\alpha_2\$ \cdots \alpha_m\$$，其中各个块 $\alpha_i$,

$i = 1, \ldots, m$是 $\mathcal{T}$配置的编码，而 \$ $\notin Q \cup \Gamma$ 是分隔符。因此， $\mathcal{T}$在输入 $w$ 上的有效计算（的编码）是一个字符串 $\alpha_1\$\alpha_2\$ \cdots \alpha_m\$$，其中某个整数 $m \geq 1$ 满足：

1. $\alpha_i \in \Gamma^* Q \Gamma^*$，即 $\alpha_i$编码 $\mathcal{T}$的一个配置， $i = 1$，.....， $m$；2. $\alpha_1$ 编码输入 $w$ 上的初始配置，所用字符串为 $q_I w$，其中 $q_I$ 是 $\mathcal{T}$的初始状态；3. $\alpha_m$ 编码 $\mathcal{T}$的一个停机配置，即一个无法进行任何移动的配置；4. $\alpha_{j+1}$可从$\alpha_j$经一步到达， $j = 1$，.....， $m - 1$。

我们用Valid($\mathcal{T}$， $w$)表示 $\mathcal{T}$在输入 $w$上的有效计算集合，并用Valid($\mathcal{T}$， $w$)$^c$ 表示其补集。

**引理4** 设 $\mathcal{T}$ 是一台图灵机， $w$ 是一个固定字符串。集合Valid($\mathcal{T}$， $w$)$^c$ 由一台1转向 OCA接受。

证明。 语言Valid($\mathcal{T}$， $w$)$^c$ 由一台OCA $\mathcal{M}_{\mathcal{T},\, w}$ 按如下策略接受。给定 $\mathcal{D} = \beta_1\$\beta_2\$ \cdots \$\beta_m$，其中 $\beta_j \in (Q \cup \Gamma)^*$， $j$ =1，.....， $m$，为判定 $\mathcal{D} \in$ Valid($\mathcal{T}$， $w$)$^c$是否成立， $\mathcal{M}_{\mathcal{T},\, w}$猜测条件1、2、3和4中哪一个未被满足。对于前三个条件，对猜测的验证仅使用有限控制完成。对于最后一个条件，$\mathcal{M}_{\mathcal{T},\, w}$ 可以猜测一个块 $j$、 $1 \leq j < m$，以及块 $j$与 $j+1$之间的一个“不匹配”位置。验证时使用计数器来定位这两个块中的相应位置。

□

现在我们可以证明本节的主要结果：

**定理 1** 以下性质是不可判定的：

- 某个 OCA 是否能在有限次转向中接受，
- 对于给定的整数 $k > 0$，某个 OCA 是否能在 $k$ 轮次内接受。

证明。 我们给出从空白带停机问题出发的归约，即判定一台图灵机在空白带上是否停机的问题。

给定一台图灵机 $\mathcal{T}$，其状态集为 $Q$，工作字母表为 $\Gamma$，使得 $\$ \notin Q \cup \Gamma$，我们考虑语言

$$L = \{x\$y \mid x \in \{a,b\}^*, y \in (Q \cup \Gamma \cup \{\$\})^*, \text{ and } (x \in \mathsf{Eq}^* \text{ or } y \notin \mathsf{Valid}(\mathcal{T}, \varepsilon))\}.$$

我们构造一个 OCA $\mathcal{M}$，它接收形式为 $x\$y$ 的输入 $z$，其中 $x \in \{a,b\}^*$，并通过非确定性地选择以下两种方式之一来判定 $z \in L$：验证 $x \in \mathsf{Eq}^*$（通过模拟 OCA $\mathcal{A}_{\mathsf{Eq}^*}$ 来完成），或验证 $y \notin \mathsf{Valid}(\mathcal{T}, \varepsilon)$（通过模拟一个 1 转向的 OCA 接受 $\mathsf{Valid}(\mathcal{T}, \varepsilon)^c$ 来完成）。

如果 $\mathcal{T}$ 在输入 $\varepsilon$ 上停机，那么固定一个 $\hat{}y \in \mathsf{Valid}(\mathcal{T}, \varepsilon)$。对于每个 $k > 0$，在输入 $(ab)^k\$\hat{}y$ 上唯一的接受计算会使用 $k$ 转向。因此，$\mathcal{M}$ 为接受而使用的转向次数不可能被任何常数所界定。

然而，如果 $\mathcal{T}$ 在输入 $\varepsilon$ 上不停机，那么 $\mathsf{Valid}(\mathcal{T}, \varepsilon)$ 为空，因此，每个满足 $x \in \{a,b\}^*$ 的输入 $z = x\$y$ 都会被 $\mathcal{M}$ 接受；其计算通过验证 $y \in \mathsf{Valid}(\mathcal{T}, \varepsilon)^c$，至多使用一次转向。

因此，我们可以得出结论：$\mathcal{M}$ 在有限次转向内接受，当且仅当 $\mathcal{T}$ 在输入 $\varepsilon$ 上不停机，而这是不可判定的。所以，无法判定一个 OCA 是否在有限次转向内接受。

第二条陈述源于如下事实：当上述机器 $\mathcal{M}$ 在有限次转向内接受时，它便在一次转向内接受，因而对每个 $k > 0$都在 $k$ 轮次内接受。

□

现在我们证明，若限制为 $k = 0$，则定理 1中所考虑的第二条性质即使在 PDA的情形下也变得可判定。我们指出，在0轮次内的接受意味着下推存储器是无用的，因此该机器所接受的语言是正则的。然而，我们所研究的性质不同于判定所接受的语言是否为正则，后者是不可判定的 [BHPS61]。事实上，存在一些 PDA，即使使用轮次，也会接受正则语言。

**定理 2** 一个 PDA $\mathcal{M}$ 是否在0 轮次内接受是可判定的。

证明。设 $\mathcal{M}$ 的下推存储器的起始符号为 $\perp$。我们可以假设，只有通过 $\varepsilon$-转移才能移除 $\perp$，因此恰好在扫描完所有输入时接受（例如，可以使用引理 2 中的构造来实现这一点）。

从 $\mathcal{M}$ 中移除所有执行压入或弹出操作的转移，并将 $\mathcal{M}$ 中所有会从栈中弹出 $\perp$ 的状态选作终态，我们便得到一个有限自动机 $\mathcal{A}$，它接受所有无需任何转向即可被 $\mathcal{M}$ 接受的字符串。于是，$\mathcal{M}$ 在 0 轮次接受，当且仅当 $L(\mathcal{A}) = L(\mathcal{M})$；而由于 $L(\mathcal{A}) \subseteq L(\mathcal{M})$，这等价于 $L(\mathcal{M}) \cap (L(\mathcal{A}))^c = \emptyset$。由于上下文无关语言对与正则语言取交集有效封闭，且上下文无关语言的空性是可判定的，我们得出该性质是可判定的。

□

本节的最后一部分专门探讨规模权衡。

**定理 3** 存在一个函数 $T : \mathbb{N} \to \mathbb{N}$，它无法被任何递归函数界定，并且对无穷多个整数 $s$，存在一个上下文无关语言 $L_s$，它具有大小为 $s$的描述，满足：

- 存在一个在 $T(s)$ 轮次内接受的 OCA $L_s$。[8]
- 对于无穷多个整数 $n$，每个接受型 PDA $L_s$ 至少使用 $T(s)$ 轮次来接受长度为 $n$的字符串。

证明。该论证源自 [MF71，命题 7]。我们先介绍 $\mathsf{Valid}(\mathcal{T}, w)$ 的补集的一种如下修改：其中每个分隔符 \$ 都替换为 $\{a, b\}^+$ 中的一个字符串。出于技术原因且不失一般性，假设 $a, b \notin Q \cup \Gamma \cup \{\$\}$，其中，照例，$Q$ 是状态集，$\Gamma$ 是工作字母表，而 $\$ \notin Q \cup \Gamma$：

$$
\begin{aligned}
\mathsf{pNotValid}(\mathcal{T}, w) \;=\; \{&\alpha_1 z_1 \alpha_2 z_2 \cdots \alpha_m z_m \mid m > 0, \\
&\alpha_i \in (\Gamma \cup Q)^+, z_i \in \{a, b\}^+,\ i = 1, \ldots, m, \text{ and:} \\
&\text{(a) } \alpha_1 \$ \alpha_2 \$ \cdots \alpha_m \$ \in \mathsf{Valid}(\mathcal{T}, w)^c, \text{ or} \\
&\text{(b) } \forall i, 1 \le i \le m,\ z_i \in \mathsf{Eq} \}\,.
\end{aligned}
$$

通过调整用于证明定理 1 的论证，我们定义一个 OCA $\mathcal{M}$，它采用以下策略接受 $\mathsf{pNotValid}(\mathcal{T}, \varepsilon)$。在计算开始时，对于输入 $x = \alpha_1 z_1 \alpha_2 z_2 \cdots \alpha_m z_m$，$\mathcal{M}$ 非确定性地

[8]我们指出，参数 $s$ 是语言 $L_s$描述的大小。因此，$T(s)$ 相对于输入长度是常数。这意味着，对于 $L_s$ 中的每个字符串，OCA 都有一条至多进行 $T(s)$ 次转向的接受计算。

决定验证条件 (a) 和条件 (b) 之一。按照定理 1 的证明中所述，最多只需一次转向即可检验条件 (a)。为检验条件 (b)，扫描输入的同时，在每个块 $z_i$ 上模拟 OCA $\mathcal{A}_{\mathsf{Eq}}$。这恰好需要 $m$ 轮次。我们指出， $\mathcal{M}$ 的描述大小关于给定机器 $\mathcal{T}$ 的状态数是多项式级的。

考虑任意一台停机单带确定性图灵机 $\mathcal{T}_s$，其描述大小为 $s$。令 $\mathcal{C}_s$ 为 $\mathcal{T}_s$ 在输入 $\varepsilon$上的计算的编码。众所周知， $\mathcal{C}_s$ 的长度无法由 $\mathcal{T}_s$ 的状态数以及 $s$中的任何递归函数所界定。

我们将按照前述构造得到的 OCA 接受 $\mathcal{M}_s$ 的 pNotValid($\mathcal{T}_s$, $\varepsilon$) 记为 $\mathcal{M}_s$ 。

假设 $\mathcal{T}_s$ 在 $\varepsilon$上停机。由于 $\mathcal{T}_s$ 是确定性的，在 $\varepsilon$上的计算 $\mathcal{C}_s$ 只有一种。此计算由字符串 $\alpha_1\$\alpha_2\$\cdots\alpha_m\$ \in$Valid($\mathcal{T}_s$, $\varepsilon$) 描述。这意味着，对于每个满足 $z_i \in$Eq 的字符串 $x = \alpha_1 z_1 \alpha_2 z_2 \cdots \alpha_m z_m$，其中 $i = 1$、.....、 $m$，都满足唯一的 条件 (b)。因此，无论长度如何，都需要 $m$ 轮次才能被接受。我们取 $T(s) = m$。由于 $\mathcal{C}_s$ 中出现的配置数量 $m$ 无法由 $s$中的任何递归函数所界定，因此函数 $T(s)$无法由 $s$中的任何递归函数所界定。

为证明接受 $L_s$所需的转向次数下界（由任意 PDA），我们再次考虑计算 $\mathcal{C}_s$在 $\varepsilon$上的编码 $\alpha_1\$\alpha_2\$\cdots\alpha_m\$$。 $\alpha_1\{a,b\}^*\alpha_2\{a,b\}^*\cdots\alpha_m\{a,b\}^*$ 与 pNotValid($\mathcal{T}_s, \varepsilon$) 的交集恰好是 $\alpha_1$Eq$\alpha_2$Eq $\cdots$ Eq$\alpha_m$Eq。因此，根据引理 3，对于每个 PDA 接受 pNotValid($\mathcal{T}_s, \varepsilon$)，识别形如 $\alpha_1 z \alpha_2 z \cdots \alpha_m z$且满足 $z = a^K b^K$的任意输入，对于每个充分大的 $K$，都需要 $m$ 次转向。

□

与定理 3 不同，通过考虑所有接受计算，我们可以得到一个递归下界：

**定理 4** 存在一个指数函数 $f$，使得如果 $\mathcal{M}$ 是一个 $k$转向的 PDA，那么 $k \leq f(s)$，其中 $s$ 是 $\mathcal{M}$ 的大小。

证明。 我们调整并完善用于证明判定 PDA是否为有限转向 [GS66（定理 5.1]) 的构造。给定一个 PDA $\mathcal{M}$，我们可以对其进行修改，以便在其有限控制器中跟踪当前的“阶段”

即下推存储器是递增还是递减。为此，将状态 $Q$ 集合扩展一个 $\{\uparrow,\downarrow\}$分量，并在初始状态中将其设为 $\uparrow$。模拟弹出操作时，如果第二个分量为 $\uparrow$，则将其切换为 $\downarrow$。相反，执行压入操作时，如果第二个分量为 $\downarrow$，则将其改为 $\uparrow$。其他情况下，第二个分量保持不变。所得 PDA 与原来的等价。

现在，我们进一步修改该机器，将转移中的输入符号替换掉，从而得到一个 PDA，其字母表为一元字母表 $\{a\}$，并在输入中计数转向次数。每个对应一次转向的转移，也就是第二个分量从 $\uparrow$变为 $\downarrow$的转移，都会从输入中读取符号 $a$；所有其他转移则变为 $\varepsilon$-转移。这样，所得的 PDA $\mathcal{M}_a$ 接受以下语言：

$$L_a = \{a^t \mid \mathcal{M} \text{ 作为包含 } t \text{ 次转向的接受计算}\}.^{9}$$

根据 [PSW02], 中的 $\mathcal{M}_a$，我们可以得到一个非确定性有限自动机 $A$，它接受 $L_a$，其状态数不超过 $f(s)$，其中 $f$是一个指数函数。

如果 $\mathcal{M}$ 是 $k$-转向的，那么 $L_a$ 是有限的，且只包含长度不超过 $k$的字符串。因此，自动机 $A$ 至少应包含 $k+1$ 个状态。由此可得 $k < f(s)$。

□

由定理3和定理4可得：

**推论1** OCA 与 PDA 在常数次转向下接受的语言和有限转向的 PDA 之间的权衡不可递归。

## 4 次线性转向

在本节中，我们介绍一种语言 $\mathrm{L}_{\mathrm{sq}}$，它可以通过次线性转向次数被接受。该语言是 [Gab84], 中所介绍语言的“填充”版本，其中长度为 $n$ 的字符串可以由一个 OCA 使用

[9]事实上，由于接受方式为空栈接受，在修改后的 $\mathcal{M}$ 中，0 次转向的接受计算由一系列配置组成，这些配置中状态的第二个分量均为 $\uparrow$；随后弹出栈底符号 $Z_0$，最终到达状态的第二个分量为 $\downarrow$的配置。因此，$L_a$ 中的字符串 $a^1$ 可以表示进行 0 次或 1 次转向的计算。

计数器上界为 $O(\sqrt{n})$ 的计数器自动机接受。我们将证明，$\mathsf{L_{sq}}$ 可以由一个 OCA 接受，该自动机使用 $O(\sqrt{n})$ 次转向。此外，即使使用 a PDA，这一界限也无法得到显著改进。

语言 $\mathsf{L_{sq}}$ 定义在字母表 $\{0, a, b\}$ 上。粗略地说，如果字符串的极大因子在 $0^+$ 中的序列与从 1 到某个 $m \geq 1$ 的所有整数的一元表示序列不一致，或者它在 $\{a, b\}^+$ 中的所有极大因子都属于语言 Eq，那么该字符串就属于该语言：

$\mathsf{L_{sq}} = \{x_1 z_1 x_2 z_2 \cdots x_m z_m \mid m > 0, x_i \in \{0\}^+, z_i \in \{a, b\}^+, i = 1, \ldots, m,$ 以及：(a) $\exists i, 1 \leq i \leq m, |x_i| \neq i,$ 或 (b) $\forall i, 1 \leq i \leq m,\ z_i \in \mathsf{Eq}\}$.

现在我们证明，$\mathsf{L_{sq}}$ 可由一个 OCA 在数量级为 $\sqrt{n}$ 的转向次数内接受。此外，我们证明了一个数量级为 $\sqrt[3]{n}$ 的下界，该下界在 PDAs 的情况下也成立。

**定理 5** 语言 $\mathsf{L_{sq}}$ 由一台 OCA 在 $O(\sqrt{n})$ 轮次内接受。此外，每台 PDA 都不能使用 $o(\sqrt[3]{n})$ 轮次接受 $\mathsf{L_{sq}}$。

证明。给定一个字符串 $w = x_1 z_1 x_2 z_2 \cdots x_m z_m$，一台 OCA $\mathcal{A}$ 可以通过非确定性地从条件 (a) 和 (b) 中选择一个并进行验证，来判定 $w \in \mathsf{L_{sq}}$。（我们假设 $w$ 的格式正确。扫描 $w$时，可以通过有限控制来检查这一点。）

我们注意到，条件 (a) 成立当且仅当 $x_1 \neq 1$，或者存在 $i, 1 \leq i < m$，使得 $|x_{i+1}| \neq |x_i| + 1$。因此，可以通过检查 $x_1 \neq 1$，或者非确定性地选择一个块 $x_i$，将 $|x_i|$保存到计数器中，并在进行一次转向后，使用计数器验证下一个块（如果存在）的长度，来检验该条件。如果验证成功，机器就接受。

条件 (b) 可通过对每个区块 $z_i$ 模拟 OCA $\mathcal{A}_{\mathsf{Eq}}$接受等式来验证。每个 $z_i$ 需要一次转向，因此总共需要 $m$轮次。

由于接受满足 (a) 的字符串只需一轮，因此，为了得到转向次数的总数，我们估算任意满足仅 条件 (b) 的字符串 $w$（其长度为 $n$）对应的块数 $m$，即 $w = 0^1 z_1 0^2 z_2 \cdots 0^m z_m$，其中 $m > 0$ 和 $z_1, z_2, \ldots, z_m \in \mathsf{Eq}$。这给出 $n = |w| > \sum_{i=1}^{m} i = m(m+1)/2$。因此，接受 $w$ 所用的转向次数 $m$ 为 $O(\sqrt{n})$。

为证明下界，考虑一个接受 $L_{sq}$ 的 PDA $\mathcal{M}$。令 $N$为 $\mathcal{M}$对应的引理 3 中的常数。给定一个整数 $m > 0$，根据 $L_{sq}$ 的定义，$0^1\{a,b\}^*0^2\{a,b\}^* \cdots 0^m\{a,b\}^*$ 与 $L$的交集为 $0^1Eq0^2Eq \cdots 0^mEq$。令 $N' = N \cdot (m(m+1)/2+1)$。根据引理 3，要被 $\mathcal{M}$接受，字符串 $w = 0^1a^{N'}b^{N'}0^2a^{N'}b^{N'} \cdots 0^ma^{N'}b^{N'}$需要 $m$ 次转向。此外 $n = |w| = {}^{m}{}_{(m+1)}{}_{2} + 2N'm = \Theta(m^3)$。因此，对每个$\sqrt{m}$，都存在一个输入长度 $n$，使得需要 $m$ 次转向，其中 $m = \Theta(\sqrt[3]{n})$。由此可得，$\mathcal{M}$ 无法通过 $o(\sqrt[3]{n})$ 次转向来接受。 □

## 5 一个无限层级

在第 4 节中，我们介绍了一种语言 $L_{sq}$，它可在次线性转向次数内被接受，且该次数不会低于 $\sqrt[3]{n}$。现在我们将证明，存在一些语言，其所需且充分的转向次数可以大幅减少。具体而言，我们将证明，对于每个整数 $k$，都存在一种语言 $L_k$，它可在数量级为 $\lg^{(k)} n$ 的转向次数内被接受，而且这也是必要的。这就构成了关于转向次数的无限层级。

为得到这一结果，我们先介绍一些预备工具。我们用 $\mathsf{bin}(x)$ 表示任意整数 $x > 0$ 的二进制表示。众所周知，$|\mathsf{bin}(x)| = \lfloor \lg x \rfloor + 1$。

给定一个形如 $y = x_1\$x_2\$ \cdots x_m\$$ 的字符串 $y$，其中 $x_i \in \{0,1\}^+$，$i = 1,\ldots,m$，我们用 $\mathsf{last}(y)$ 表示 $y$ 中最右侧的二进制块，即块 $x_m$。

我们考虑字母表 $\{0,1,\$\}$ 上的以下语言：

$$\mathsf{ListBin} = \{\mathsf{bin}(1)\$\mathsf{bin}(2)\$ \cdots \$\mathsf{bin}(m)\$ \mid m \geq 1\}.$$

ListBin 中的字符串是从 1 到某个 $m \geq 1$ 的二进制编码整数列表，其中每个编码都以分隔符 $ 结尾。

通过抽引论证可以证明，ListBin 不是上下文无关语言。然而，它的补集是上下文无关的。更准确地说，它可由一个 1-转向 OCA [LSH65、Gab84] 接受。

滥用记号，对于 $y \in$ListBin，我们也用 $\mathsf{last}(y)$ 表示 $y$中最右侧二进制块所代表的整数。

**引理 5** 若 $y = x_1\$x_2\$\cdots\$x_m\$ \in \mathsf{ListBin}$，且 $x_i \in \{0,1\}^*$、$i = 1$、……、$m$，则：

- $|\mathsf{last}(y)| \le \lg |y|$，
- $|y|_\$ = m = $最后$(y) \ge 2^{|\text{最后}(y)|-1}$，
- $|y| \le 2m + m\lg m$。

证明。由于 $y \in \mathsf{ListBin}$，$x_i = \mathsf{bin}(i)$ 对于 $i = 1$，……，$m$。此外，由于分隔符，$m \le |y|/2$。因此：

$$|\mathsf{last}(y)| = |x_m| = |\mathsf{bin}(m)| \le 1 + \lg m \le 1 + \lg \frac{|y|}{2} = \lg |y| .$$

第二个陈述中的等式显然成立。此外，$|\mathsf{last}(y)| = |\mathsf{bin}(m)| = \lfloor \lg m \rfloor + 1$。因此，$2^{|\mathsf{last}(y)|-1} = 2^{\lfloor \lg m \rfloor} \le m$。

对于第三个陈述，我们注意到，对于 $m > 0$，$\sum_{i=1}^{m} \lfloor \lg i \rfloor \le m\lg m$。因此：

$$|y| = \sum_{i=1}^{m} |\mathsf{bin}(i)\$| = \sum_{i=1}^{m} (\lfloor \lg i \rfloor + 2) \le 2m + m \lg m .$$

□

在介绍所述层级之前，我们先证明：给定一个接受语言 $L \subseteq \Sigma^*$、具有 PDA $O(t(n))$ 轮次的 PDA，总能构造一个接受另一个语言、使用 $O(t(\log n))$ 轮次的 PDA。更准确地说，我们考虑语言 的扩展版本 $L$，记作 $\mathsf{Ext}(L)$。它是字母表 $\Delta = \Sigma \cup \{0,1,\$\}$ 上的语言，定义如下：[10]

$$\begin{aligned}\mathsf{Ext}(L) = \{x_1\$x_2\$\cdots\$x_m\$\$x \quad | \quad & x_i \in \{0,1\}^+, i = 1,\ldots,m,\ x \in \Delta^*, \\ & \text{and at least one of the following holds:} \\ & \text{(a) } x_1\$x_2\$\cdots\$x_m\$ \notin \mathsf{ListBin}, \\ & \text{(b) } |x_m| < |x|, \\ & \text{(c) } x \in L \ \}.\end{aligned}$$

我们指出，在上述定义中，如果 $x_1\$x_2\$\cdots\$x_m\$ \notin \mathsf{ListBin}$，即满足条件 (a)，那么后缀 $x$ 可以是 $\Delta^*$ 中的任意字符串。

[10]请注意，我们不要求 $\Sigma \cap \{0,1,\$\} = \emptyset$。这样就可以迭代地定义扩展版本的扩展版本，而无需扩大输入字母表。

否则，如果 $|x_m| \geq |x|$ （即不满足条件 (b)），那么 $x$必须属于 $L$ （条件 (c)）。仅满足条件 (c) 的字符串对于得出接下来的结果至关重要。

**定理 6** 设语言 $L$ 由一个在 $t(n)$ 轮次内接受它的 PDA $\mathcal{M}$所接受，则存在一个 PDA $\mathcal{M}'$ 接受器，可接受 $\mathsf{Ext}(L)$，并使用 $t'(n)$ 轮次，其中 $t'(n) \leq \max(1, t(\lg(n)))$。此外，如果 $\mathcal{M}$是一个 OCA ，那么 $\mathcal{M}'$ 也会是一个 OCA。

证明。 根据接受 PDA $\mathcal{M}$ 的 $L$，我们为 $\mathsf{Ext}(L)$ 定义一个 PDA $\mathcal{M}'$： 对于形如 $x_1\$x_2\$ \cdots \$x_m\$\$x$ 的输入 $w$，它会非确定性地从 $\mathsf{Ext}(L)$ 的定义中的条件（a）、（b）、（c）中选择一个，并对其进行检验。

条件（a）和（b）可以通过将下推存储器用作计数器，并且至多进行一次转向来检验。因此，当 $w$ 满足其中一个条件时，一次转向就足够了。条件（c）通过在后缀 $x$ 上模拟给定的 PDA $\mathcal{M}$来检验，至多使用 $t(|x|)$ 轮次。如果 $w$ 仅满足（c），那么根据引理 5， $|x| \leq |x_m| \leq \lg |x_1\$x_2\$ \cdots \$ x_m\$| \leq \lg |w|$。因此，在这种情况下， $t(\lg |w|)$ 轮次就足够了。

由于在检查条件 (a) 和 (b) 时，下推存储器被用作计数器，因此我们很容易得出结论：如果 $\mathcal{M}$ 是一个 OCA ，那么 $\mathcal{M}'$ 也是一个 OCA。 □

现在来考察 $(\mathsf{Eq}\$)^+$。容易看出，这种语言可由一个使用线性数量转向次数的 OCA 接受。因此，根据定理 6，对于每个 $k \geq 0$，语言 $\mathsf{Ext}^k((\mathsf{Eq}\$)^+)$ 可由一个在 $\max(1, \lg^{(k)} n)$ 轮次内的 OCA 接受。

现在，我们对语言 $\mathsf{Ext}^k((\mathsf{Eq}\$)^+)$ 稍作修改，以获得所述层级。更具体地说，对于每个 $k \geq 0$，我们定义以下关于 $\Sigma = \{a, b, 0, 1, \$\}$ 的语言：[11]

$$
\begin{aligned}
\mathsf{L}_k = \{ y_k\$ \cdots \$ y_1\$ y_0 \quad | \quad & y_i \in \{\{0,1\}^+\$\}^+, i = 1, \ldots, m, y_0 \in \{a, b, \$\}^*, \\
& \text{and at least one of the following holds:} \\
& \quad (\mathrm{a})\ \exists i, 1 \leq i \leq k, y_i \notin \mathsf{ListBin}, \\
& \quad (\mathrm{b})\ \exists i, 1 \leq i \leq k, |\mathsf{last}(y_i)| \leq |y_{i-1}|_\$, \\
& \quad (\mathrm{c})\ y_0 \in (\mathsf{Eq}\$)^+ \} .
\end{aligned}
$$

[11]为了使技术性表述更为简明，本文此处介绍的语言 $\mathsf{L}_k$ 不同于论文初稿版本 [Pig25] 中的语言。不过，论证和所用技术相似。

$L_k$ 与 Ext$^k$((Eq\$)$^+$) 之间最重要的区别在于条件 (b)。采用这一条件的新版本后，我们仍能证明，接受 $L_k$ 中字符串所需的转向次数存在一个 $\lg^{(k)} n$ 的上界；同时，我们也将能够证明，确实需要这一数量级的转向次数。

我们注意到，要检验条件 (b)，块 $y_0$ 中的符号 $a$ 和 $b$ 无关。$y_0$ 中剩余的符号是 \$s。每个满足仅 条件 (c) 的输入字符串 $w$ 中，这些符号的出现次数与接受 $w$所用的转向次数相同。

**例 1** 我们来考虑以下字符串：

$w_1 =$ 1\$10\$11\$100\$\$1\$10\$\$$ab$\$$ab$\$$aabb$\$\$$w_2 =$ 1\$10\$11\$100\$101\$110\$111\$1000\$\$1\$10\$11\$\$$ab$\$$ba$\$$w_3 =$ 1\$10\$11\$100\$101\$110\$111\$1000\$\$1\$10\$11\$\$$aabb$\$

它们都属于语言 $L_2$，且不满足该语言定义中的条件 (a)。具体来说：

- 字符串 $w_1$ 满足条件 (b)，因为 $|$last(1\$10\$)$| = 2 \lhd |ab\$ab\$abba\$|_\$ = 3$。此外，由于为 $ab\$ab\$aabb\$ \in$ (Eq\$)$^+$，$w_1$ 也满足条件 (c)。

- 字符串 $w_2$ 满足条件 (b)，因为 $|$last(1\$10\$11\$)$| = 2 = |ab\$ba\$|_\$$，但它不满足条件 (c)。

- 字符串 $w_3$ 仅满足条件 (c)。事实上，$aabb\$ \in$ (Eq\$)$^+$，而 $|$last(1\$10\$11\$100\$101\$110\$111\$1000\$)$| = 4 > |$1\$10\$11\$$|_\$ = 3$且 $|$last(1\$10\$11\$)$| = 2 > |aabb\$|_\$ = 1$，

□

通过对用于证明定理6的论证加以细化，我们现在得到接受 $L_k$ 所需的充分转向次数的上界：

**定理 7** 对于每个 $k \geq 0$，存在一个 OCA $\mathcal{A}_k$，以 $max(1,\lg^{(k)} n)$ 轮次接受语言 $L_k$。

证明。固定 $k \geq 0$，令 $w = y_k\$ \cdots \$y_1\$y_0$ 为 $L_k$ 中的一个字符串。根据上述定义，条件 (a) 和 (b) 均可通过一次转向验证。因此，对于仅满足条件 *(c)* 的字符串，可能需要更多的转向次数。

特别地，如果 $w$ 仅满足 (c)，那么它会在 $t$轮次内被接受，其中 $t = |y_0|_{\$}$。现在我们将 $t$ 与 $w$的长度 $n$联系起来。由 (a) 的否定以及引理 5 可得，对于 $i = 1, \ldots\ldots, k$:

$$\mathsf{last}(y_i) \geq 2^{|\mathsf{last}(y_i)|-1}.$$

此外，当 $i > 1$ 时，由 (b) 的否定可得 $|\mathsf{last}(y_i)| > |y_{i-1}|_{\$}$，即 $|\mathsf{last}(y_i)| - 1 \geq |y_{i-1}|_{\$}$，从而得到

$$\mathsf{last}(y_i) \geq 2^{|\mathsf{last}(y_i)|-1} \geq 2^{|y_{i-1}|_{\$}} = 2^{\mathsf{last}(y_{i-1})}$$

因此，对于 $i = 2, \ldots, n$，有 $\lg \mathsf{last}(y_i) \geq \mathsf{last}(y_{i-1})$。出于同样的原因，$\mathsf{last}(y_1) \geq 2^{|y_0|_{\$}} = 2^t$，且 $\lg \mathsf{last}(y_1) \geq t$。由此可得，对于 $i = 1, \ldots, k$，有 $t \leq \lg^{(i)} \mathsf{last}(y_i)$。由于 $n = |w| \geq \mathsf{last}(y_k)$，最终可得 $t \leq \lg^{(k)} n$。证明至此完成。 □

现在我们将为接受 $\mathrm{L}_k$ 所需的转向次数给出一个下界。为此，我们先引入一些字符串；正如我们将要证明的那样，接受这些字符串需要数量级为 $\lg^{(k)} n$ 的转向次数。

首先，对于整数 $t$，$k \geq 0$，我们来考虑以下数:

$$n_{t,k} = \begin{cases} t & \text{if } k = 0, \\ 2^{n_{t,k-1}} & \text{otherwise.} \end{cases}$$

对于 $k > 0$，我们定义以下字符串:

$$y_{t,k} = \mathsf{bin}(1)\$\mathsf{bin}(2)\$ \cdots \$\mathsf{bin}(n_{t,k})\$$$

and

$$w_{t,k} = y_{t,k}\$y_{t,k-1}\$ \cdots \$y_{t,1}\$$$

**引理6** 如果 $t > 2$，那么 $|w_{t,\,k}| < n_{t,\,k}^2$。

证明。为证明该结论，我们使用下面的不等式，它对每个 $x > 2$ 都成立，且很容易验证:

$$2^x(2 + x) + x^2 + 1 < 2^{2x}. \tag{1}$$

我们注意到，根据引理 5，$|y_{t,\,k}| \leq 2n_{t,\,k} + n_{t,\,k} \lg n_{t,\,k}$，其中 $k > 0$。

我们通过对 $k$进行归纳，证明所声称的 $|w_{t,k}|$ 上界。

如果 $k=1$，利用带有 $x=t$ 和 $n_{t,1}=2^t$ 的（1），我们得到：

$$|w_{t,1}| = |y_{t,1}| + 1 \le 2^{t+1} + 2^t t + 1 = 2^t(2+t) + 1 < 2^{2t} = n_{t,1}^2 .$$

对于 $k>1$，利用 $|w_{t,k-1}| < n_{t,k-1}^2$ 作为归纳假设以及 (1)，我们得到：

$$\begin{aligned} |w_{t,k}| &= |y_{t,k}| + 1 + |w_{t,k-1}| \\ &< 2n_{t,k} + n_{t,k}\lg n_{t,k} + 1 + n_{t,k-1}^2 \\ &= 2\cdot 2^{n_{t,k-1}} + 2^{n_{t,k-1}}\cdot n_{t,k-1} + 1 + n_{t,k-1}^2 \\ &= 2^{n_{t,k-1}}(2+n_{t,k-1}) + n_{t,k-1}^2 + 1 \\ &< 2^{2n_{t,k-1}} = (2^{n_{t,k-1}})^2 = n_{t,k}^2 . \end{aligned}$$

□

**引理 7** 对于每个整数 $h>0$ 以及每个字符串 $x \in (式\$)^h$，字符串 $w = w_{t,k}x$ 属于语言 $\mathsf{L}_k$。此外，$w$ 满足 仅条件 (c)于 $\mathsf{L}_k$ 的定义中，当且仅当 $h \le t$。

证明。 根据定义，字符串 $w$ 不能满足条件 (a)。

关于条件 (b)，我们注意到，根据 $w_{t,k}$ 的定义，若 $i>1$，则 $|\mathsf{last}(y_i)| = \lfloor \lg n_{t,i} \rfloor + 1 = n_{t,i-1} + 1$ 以及 $|y_{i-1}|_\$ = n_{t,i-1}$，由此得到 $|\mathsf{last}(y_i)| > |y_{i-1}|_\$$。因此，条件 (b) 被满足，当且仅当 $|\mathsf{last}(y_1)| \le |y_0|_\$$。由于 $\mathsf{last}(y_1) = n_{t,1} = 2^t$，$|\mathsf{last}(y_1)| = t+1$，而 $|y_0|_\$ = h$，我们得到条件 (b) 被满足当且仅当 $t+1 \le h$，即 $h > t$。

这使我们能够得出结论：$w$ 满足条件 *(c)* 当且仅当 $h \le t$。

□

现在我们可以证明这个下界：

**定理 8** 对于每个 PDA $\mathcal{M}_k$，若其接受语言 $\mathsf{L}_k$，$k>0$，则对于无穷多个整数 $n$，存在一个长度为 $n$ 的字符串，至少需要 $\lg^{(k)} n - 2$ 轮次才能被 $\mathcal{M}_k$ 接受。

证明。 固定一个整数 $k>0$，考虑一个接受 $\mathsf{L}_k$ 的 PDA $\mathcal{M}_k$。我们要证明，为接受长度为 $n$ 的输入，$\mathcal{M}_k$ 所需的转向次数随 $\lg^{(k)} n$ 按 lg 增长。令 $N$ 为 $\mathcal{M}_k$ 对应的引理 3 中的常数。

给定 $t>2$，取 $N' = N\cdot(|w_{t,k}| + t + 1)$，并考虑字符串 $x = (a^{N'}b^{N'}\$)^t$ 和 $w = w_{t,k}x$。根据引理 7，$w \in \mathsf{L}_k$，且它

满足 $L_k$ 的定义中仅有条件 *(c)* 。现在我们证明，接受 $w$ 需要 $t$ 次转向。为此，我们注意到 $w = w_{t,k} z_1 \$ z_2 \cdots z_t \$$，其中 $z_i = a^{N'} b^{N'}$, $i = 1, \ldots, t$。由于 $w$ 仅满足条件 (c)，如果将其中的一些块 $z_i$ 替换为 $\{a,b\}^* \setminus \mathrm{Eq}$ 中的某个字符串，就会得到一个不属于 $L_k$ 的字符串；而如果替换时只使用 Eq 中的字符串，那么所得字符串属于 $L_k$。换言之，$w_{t,k}\{a,b\}^*\$ \{a,b\}^* \cdots \{a,b\}^*\$ \cap L_k = w_{t,k}\mathrm{Eq}\$\mathrm{Eq} \cdots \mathrm{Eq}\$$。因此，由引理 3 可知，接受 $w$ 需要 $t$ 次转向。

为完成证明，我们陈述 $n$ 的长度与接受它所需的转向次数 $t$ 之间的关系：

$$\begin{aligned} n &= |w| = |w_{t,k}| + |x| = |w_{t,k}| + (2N' + 1)t \\ &= |w_{t,k}| + (2N(|w_{t,k}| + t + 1) + 1)t \\ &= |w_{t,k}| \cdot \left(1 + \left(2N + \frac{2N(t+1)+1}{|w_{t,k}|}\right) \cdot t\right) \end{aligned}$$

对于足够大的 $k$，$|w_{t,k}| > 1 + \left(2N + \frac{2N(t+1)+1}{|w_{t,k}|}\right) \cdot t$。因此，由引理 1 可得：

$$\lg^{(k)} n < 1 + \lg^{(k)}(|w_{t,k}|) \leq 1 + \lg^{(k)} n_{t,k}^2 \leq 2 + \lg^{(k)} n_{t,k} = 2 + t\,.$$

因此，$t > \lg^{(k)} n - 2$。

由此可知，对于无穷多个整数 $n$，接受长度为 $n$ 的字符串所需的转向次数至少为 $\lg^{(k)} n - 2$。 □

结合定理 7 和定理 8，我们得到：

**定理 9** 对于每个 $k \geq 0$，都存在一种语言，它可由 *a* OCA 在至多 $max(1, \lg^{(k)} n)$ 轮次内接受，但无法由 *a* PDA在 $o(\log^{(k)} n)$ 轮次内接受。

证明。给定 $k \geq 0$，考虑语言 $L_k$。根据定理 7，$(1, \lg^{(k)} n)$ 轮次足以识别它。此外，$L_0 = (\mathrm{Eq}\$)^+$不能在 $o(n)$ 轮次内被接受；根据定理 8，$k > 0$ 语言 $L_k$ 至少需要 $\lg^{(k)} n - 2$ 轮次才能被 PDA接受。由于这个函数以 $\lg^{(k)} n$ 的速度增长，我们可以得出结论：$L_k$ 无法在 $o(\text{对数}^{(k)}\, n)$ 轮次内被接受。

□

作为定理 9 的推论，我们得到一个关于转向次数的无穷真分层：对于每个 $k \geq 0$，至多在对数$^{(k)}\, n$ 轮次内接受的语言类，严格包含至多在对数$^{(k+1)}\, n$ 轮次内接受的语言类。

## 6 更少的轮次

在本节中，我们提出一种语言 $\mathrm{U}_*$，它所需的转向次数少于第 5 节所讨论层级中的每种语言。事实上，我们证明，$\lg^* n$ 轮次足以识别 $\mathrm{U}_*$。我们还证明，对于无穷多个 $n$，这个次数是必要的。[12]

为了引入语言 $\mathrm{U}_*$，我们先针对每个 $k > 0$，考虑语言 $\mathrm{L}_k$ 的如下变体：

$\mathrm{U}_k = \{ y_k z_k y_{k-1} \cdots z_2 y_1 z_1 \mid \ y_i \in (\{0,1\}^+\$)^+,\ z_i \in \{a,b\}^+,\ i = 1, \ldots, k,$ 且以下条件至少有一个成立：(a) $\exists i,\ 1 \leq i \leq k,\ y_i \notin \mathrm{ListBin}$，(b) $\exists i,\ 1 < i \leq k,\ |\mathrm{last}(y_i)| \leq |y_{i-1}|_\$$，(c) $\forall i,\ 1 \leq i \leq k,\ z_i \in \mathrm{Eq}$ }.

我们观察到，每个满足 $y_0 \in (\{a,b\}^+\$)^k$ 的字符串 $w = y_k\$y_{k-1}\$ \cdots \$y_1\$ y_0 \in \mathrm{L}_k$，都可以通过将连续的 $y_i$ 块之间的分隔符替换为 $\{a,b\}^+$ 中 $y_0$ 的极大因子，变换为 $\mathrm{U}_k$ 的一个字符串。条件 (a)、(b) 和 (c) 与定义 $\mathrm{L}_k$ 的那些条件类似。因此，对 $\mathrm{U}_k$ 的识别可以用类似的方式完成，使用 $k$ 轮次。

**例 2** 如下文所讨论的，在 $\mathrm{U}_k$ 的定义中仅满足条件 (c) 的字符串在转向次数的研究中起着关键作用。我们给出这样一个字符串的例子，针对 $k = 4$。首先考虑以下字符串：

$y_1 =$ 1\$ $y_2 =$ 1\$10\$
$y_3 =$ 1\$10\$11\$
100\$

[12]即便在这种情况下，此处考虑的语言 $\mathrm{U}_*$ 也与论文 [Pig25] 的初稿版本中给出的语言略有不同。对于“新”语言 $\mathrm{U}_*$，我们能够证明需要 $\lg^* n$ 轮次（定理 11），而对于“旧”语言，我们只能证明需要无界的转向次数。

$y_4 =$ 1$10$11$100$101$110$111$1000$1001$1010$1011$1100$1101$1110$1111$10000$ 我们指出，$y_i =$二进制表示(1)$二进制表示(1)$ $\cdots$$二进制表示($2^{\uparrow\uparrow i-1}$)$，$i = 1,$ ……，4。设字符串 $u = y_4z_4y_3z_3y_2z_2y_1z_1$，其中 $z_1$，$z_2$，$z_3$，$z_4 \in \{a, b\}^+$。那么：

- 由于对于 $i = 1, \ldots,$4，$y_i \in$ ListBin $u$ 不满足条件 (a)。此外，$|y_i|$\$ = last($y_i$) $= 2^{\uparrow\uparrow i-1}$ 和 $|$last($y_i$)$| =$ lg($2^{\uparrow\uparrow i-1}$) $+ 1 >$lg($2^{\uparrow\uparrow i-1}$)。

- 由前面的等式可知，对于 $i = 2,$3,4，$|$last($y_i$)$| >2^{\uparrow\uparrow i-2} =$last($y_{i-1}$) $= |y_{i-1}|$\$。因此，条件 (b) 不成立。

- 由此可得，字符串 $u$ 属于 U$_4$ 当且仅当条件 (c) 成立，也就是说，当且仅当所有字符串 $z_i$ 都属于 Eq，其中 $i = 1, \ldots, 4$。

- 特别地，如果我们选择 $z_1 = z_2 = z_3 = z_4 = ab$，那么由此得到的字符串 $u$是 U$_4$ 中仅满足 (c) 的最短字符串。这个思路将在接下来的结果的证明中进一步展开，尤其是在定理 11 的证明中。

□

现在，我们将感兴趣的语言定义为 U$_* = \bigcup_{k\geq 0}$U$_k$。

**定理 10** 语言 U$_*$ 由一个 OCA 在 $\lg^* n$轮次内接受。

证明。给定一个输入字符串 $w = y_kz_ky_{k-1} \cdots z_2y_1z_1$、$k > 0$，其形式适当（可以在扫描输入时使用有限控制进行验证），为了检查 $w \in$U$_*$，OCA $\mathcal{B}$ 可以从条件 (a)、(b) 和 (c) 中猜测一个条件。条件 (a) 和 (b) 通过非确定性地选择一个块 $y_i$ 并使用一次转向来验证，正如定理 7 的证明中所述。为了检验条件 (c)，$\mathcal{B}$ 在每个块 $z_i$上模拟 OCA $\mathcal{A}_{\mathsf{Eq}}$，从而恰好进行 $k$ 轮次。这使我们可以得出结论：当 $w$ 仅满足 条件 (c) 时，$k$ 达到最大转向次数。

现在，我们将估算 $k$ 的大小，以仅满足 (c) 的输入 $w$ 的长度 $n$ 为参照。

对于 $i = 1, \ldots, k$，由于条件 (a) 不满足，根据引理 5，我们得到 last($y_i$) $\geq 2^{|\mathsf{last}(y_i)|-1}$。由于条件 (b) 不满足，当 $i > 1$ 时，我们还有 $|$last($y_i$)$| > |y_{i-1}|$\$ = last($y_{i-1}$)，即 $|$last($y_i$)$|-1 \geq$ last($y_{i-1}$)。因此，对于 $i > 1$，last($y_i$) $\geq 2^{\mathsf{last}(y_{i-1})}$。根据定义，last($y_1$)

应至少为 $1 = 2^{\uparrow\uparrow 0}$。由此可得，对于 $i = 1, \ldots, k$，$\mathsf{last}(y_i) \geq 2^{\uparrow\uparrow i-1}$，因此 $|y_i| > 2^{\uparrow\uparrow i-1}$。

因此，$n = |w| > |y_k| > 2^{\uparrow\uparrow k-1}$，这给出 $\lg^* n > k - 1$；由于 $\lg^* n$ 是一个整数，$\lg^* n \geq k$。提醒读者，接受 $w$ 只需 $k$ 轮次。

由此可以得出，识别 $\mathsf{U}_*$ 中长度为 $n$ 的字符串只需 $\lg^* n$ 轮次。

□

我们通过证明一个相匹配的下界来结束本节：

**定理 11** 设 $\mathcal{M}$ 是一个接受语言 $\mathsf{U}_*$ 的 PDA 。那么，对于无穷多个 $n$，存在一个长度为 $n$ 的字符串，它需要$\lg^* n$ 轮次才能被 $\mathcal{M}$ 接受。

证明。 对于每个 $i > 0$，考虑 $y_i = \mathsf{bin}(1)\$\mathsf{bin}(1)\$ \cdots \$\mathsf{bin}(2^{\uparrow\uparrow i-1})\$$。根据引理 5，

$$|y_i| \leq 2 \cdot 2^{\uparrow\uparrow i-1} + 2^{\uparrow\uparrow i-1} \lg(2^{\uparrow\uparrow i-1}) = 2^{\uparrow\uparrow i-1}(2 + \lg(2^{\uparrow\uparrow i-1}))\,.$$

设 $N$ 为引理 3 中关于 $\mathcal{M}$ 的常数。对于 $k > 0$，令 $N_k = N \cdot k \cdot |y_k|$，并考虑字符串

$$u_k = y_k a^{N_k} b^{N_k} y_{k-1} a^{N_k} b^{N_k} \cdots y_1 a^{N_k} b^{N_k}\,.$$

显然，字符串 $u_k$ 不满足 $\mathsf{U}_*$ 定义中的条件 (a)。此外，对于 $i = 2$，……，$k$:

$$|\mathsf{last}(y_i)| = |2^{\uparrow\uparrow i-1}| = \lg 2^{\uparrow\uparrow i-1} + 1 = 2^{\uparrow\uparrow i-2} + 1 = \mathsf{last}(y_{i-1}) + 1 > |y_{i-1}|_{\$}\,.$$

因此，字符串 $u_k$ 仅满足 $\mathsf{U}_*$ 定义中的条件 (c)。此外，如果 $k$ 足够大，那么 $N_k \geq N \cdot (|y_k \cdots y_1| + 1)$，并且通过将 $u_k$ 中的一些块 $a^{N_k} b^{N_k}$ 替换为 $\{a, b\}^*$中的字符串，我们得到的字符串属于 $\mathsf{U}_*$，当且仅当替换中使用了来自 Eq 的字符串。由引理 3 可得，接受 $u_k$需要 $k$ 轮次

为完成证明，我们将把 $k$ 与 $u_k$的长度 $n$ 联系起来。利用引理 5，我们观察到：

$$\begin{aligned} n = |u_k| &\leq k \cdot |y_k a^{N_k} b^{N_k}| \\ &= k \cdot (|y_k| + 2Nk \cdot |y_k|) = |y_k| \cdot (1 + 2Nk) \cdot k \\ &\leq 2^{\uparrow\uparrow k-1} \cdot (2 + \lg(2^{\uparrow\uparrow k-1})) \cdot (1 + 2Nk) \cdot k\,. \end{aligned}$$

如果 $k$ 足够大，那么我们有 $2^{\uparrow\uparrow k-1} > (2+\lg(2^{\uparrow\uparrow k-1}))\cdot(1+2Nk)\cdot k$。利用引理 1，我们得到：

$$\lg^{(k-1)} n = \lg^{(k-1)} |u_k| \leq 1 + \lg^{(k-1)}(2^{\uparrow\uparrow k-1}) = 2\,.$$

因此，$\lg^{(k)} n \leq 1$ 得到 $k \geq \lg^* n$。 □

# 7 结论

在近期一些关于下推自动机和单计数器自动机的论文中，我们考察了两种复杂度度量：下推存储器高度和压入复杂度 [Pig26、PP25、PP23]。在本文中，我们延续了这一研究方向，考察了转向复杂度。

虽然前两种度量之间存在直接关系，但尚不清楚转向复杂度是否在某种程度上与另外两种度量相关。上述论文中所举的、使用最小非恒定 $O(\log\log n)$ 高度或压入复杂度的 PDA 示例，在每次接受计算中至多进行一次转向。若存在使用“较小”但无界的转向次数，同时具有 $O(\log\log n)$ 高度或压入复杂度的机器，将会很有意思；或者，使用无界转向次数是否意味着需要更大的高度和更多的压入操作。我们指出，在定理 7 的证明中描述的、 OCA $\mathcal{A}_k$ 接受语言 $\mathrm{L}_k$ 的机器，可以使用无界高度来识别仅满足条件 (c) 的字符串。研究一种同时考虑高度（或压入操作次数）和转向次数的复杂度度量，将会很有意义。

另一个问题涉及能否用文法来重新表述我们的结果。已知有限转向下推自动机对应于超线性文法以及非终结符有界文法 [GS66]。假设有一个 PDA $\mathcal{M}$ 在由非常数函数 $f(n)$所界定的转向次数内接受。我们能否将 $f(n)$ 与通过“标准”变换从 $\mathcal{M}$得到的上下文无关文法上的某种度量联系起来?

最后，我们提出将研究限制在一元语言（即在单字母表上定义的语言）或有界语言情形下的问题。我们认为，在这些情形下，所有性质

在第3节中研究的性质都将变为可判定。此外，第4、5、6节中用于减少转向次数的技术似乎并不适用于这些情形。我们猜想，在这些情形下，转向次数不可能不是常数但又能像一般情形那样任意缓慢地增长。

## 参考文献

[BHPS61]Yehoshua Bar-Hillel、Micha Perles 和 Eliyahu Shamir。简单短语结构文法的形式性质。*Zeitschriftf¨urPhonetik, Sprachwissenschaft undKommunika-tionsforschung*，14：143–172，1961。[Gab84] Joaquim Gabarr´o。下推空间复杂度及相关的完整 AFL。载于 Max Fontet 和 Kurt Mehlhorn 编，《*STACS 84*，计算机科学理论方面研讨会，法国巴黎，*1984*年*4*月*11*日至*13*日，论文集》，计算机科学讲义第166卷，第250–259页。Springer，1984。[GS66] Seymour Ginsburg 和 Edwin H. Spanier。有限转向下推自动机。*SIAM*控制期刊，4(3)：429–453，1966。[Har67] Juris Hartmanis。上下文无关语言与图灵机计算。载于计算机科学的数学方面，应用数学研讨会论文集第19卷，第42–51页。美国数学学会，1967。[Har78] M. A. Harrison。《形式语言理论导论》。Addison-Wesley Longman Publishing Co., Inc.，美国马萨诸塞州波士顿，1978。[HU79] J. E. Hopcroft 和 J. D. Ullman。《自动机理论、语言与计算导论》。Addison-Wesley，1979。[Kut05] Martin Kutrib。非递归权衡现象。*Int.J. Found. Comput. Sci.*，16(5)：957–973，2005。[LSH65] Philip M. Lewis II、Richard Edwin Stearns 和 Juris Hartma-nis。识别上下文无关语言和上下文-

相关语言的内存界。载于第六届开关电路理论与逻辑设计年度研讨会，美国密歇根州安娜堡，*1965*年*10*月*6*日至*8*日，第191–202页。IEEE计算机学会，1965。[Mal07] Andreas Malcher。有限转向下推自动机之间的递归与非递归权衡。自动机、语言与组合学杂志，12(1-2)：265–277，2007。[MF71] Albert R. Meyer 和 Michael J. Fischer。自动机、文法和形式系统的描述简约性。载于第十二届开关与自动机理论年度研讨会，美国密歇根州东兰辛，*1971*年*10*月*13*日至*15*日，第188–191页。IEEE计算机学会，1971。[MP13] Andreas Malcher 和 Giovanni Pighizzini。有界上下文无关语言的描述复杂度。信息与计算，227：1–20，2013。[Pig25] Giovanni Pighizzini。上下文无关语言、下推自动机和单计数器自动机的转向复杂度。载于 Sang-Ki Ko 和 Florin Manea 编，《语言理论发展——第*29*届国际会议 *DLT 2025*，韩国首尔，*2025*年*8*月*19*日至*22*日，论文集》，计算机科学讲义第16036卷，第77–91页。Springer，2025。[Pig26] Giovanni Pighizzini。压入复杂度：最优界与可判定性。理论计算机科学，1071：115839，2026。[PP23] Giovanni Pighizzini 和 Luca Prigioniero。下推自动机与常数高度：可判定性和界。信息学学报，60(2)：123–144，2023。[PP25] Giovanni Pighizzini 和 Luca Prigioniero。下推自动机与单计数器自动机：常数与非常数内存使用量。信息与计算，306：105329，2025。[PSW02] Giovanni Pighizzini、Jeffrey Shallit 和 Ming-wei Wang。一元上下文无关文法与下推自动机、描述复杂度及辅助空间下界。*J. Comput. Syst.Sci.*，65(2)：393–414，2002。