arXiv:2301.12556v1 [cs.LO] 29 Jan 2023

λ 演算中图灵机的对数敏感编码

Beniamino Accattoli[1], Ugo Dal Lago[2], 和 Gabriele Vanoni[2]

[1]Inria & LIX, ´法国综合理工学院，UMR 7161，法国[2]博洛尼亚大学 & Inria 索菲亚-安提波利斯，意大利

摘要

本文对 Dal Lago 和 Accattoli 的图灵机在 λ 演算中的参考编码进行了修改λDLA17], 。该编码针对时间效率进行了优化，而本文的修改使其能够适用于对数空间。主要有两处变化：图灵机现在有 two 条带，即一条输入带和一条工作带；此外，输入带采用了不同的编码方式，因为参考编码在管理带时会带来线性空间开销，这对于研究对数空间而言过高。

1 引言

本文介绍一种将图灵机编码到 λ 演算中的新方法，并证明其正确性。该方法基于 Dal Lago 和 Accattoli 对单带图灵机的参考编码 [DLA17]。这种新编码专为研究对数空间复杂度而设计，但本文不展开这项研究，而是由一篇配套论文进行。本文旨在给出该编码的形式化定义，并通过繁琐的计算证明其正确性。该编

y p码的新要　　g 相较于参p考编码的不同之处在于：

- 对数敏感性：参考编码无法处理对数空间复杂度，因为它基于单带图灵机。对数空间需要使用带有只读输入带（输入字符串为 i）和普通工作带的图灵机，并且只计算工作带所用的空间——若将 i 的空间也计算在内，就不可能只使用 $\mathcal{O}(\log|i|)$ 空间。我们将这种磁带分离称为对数敏感性。相反，参考编码是对数不敏感的。

- 输入带采用不同的编码：仅将参考编码调整为适用于两条带，并不足以保留对数空间，因为参考编码是针对时间优化的：它能在 $\mathcal{O}(1)$ 时间内从带上读取内容，但其读取机制会带来 $\mathcal{O}(|i|)$ 的空间开销，而处理输入带时，空间开销必须至多为 $\mathcal{O}(\log|i|)$。因此，我们改变输入带的编码和读取机制，以空间换时间，从而以 $\mathcal{O}(|i|\log|i|)$ 时间读取，同时将空间开销控制在 $\mathcal{O}(\log|i|)$。其思路是用一个指针表示读写头的位置，该指针由位置索引给出，而

位置索引的长度为对数级；读取输入时，需要沿输入带依次扫描，直到到达该位置索引所指的位置。

- 时间复杂度不同：通过以空间换时间，这种新编码速度较慢。如果图灵机 $\mathcal{M}$ 在输入字符串 i 上运行需要 $T_{\mathcal{M}}(|i|)$ 时间，那么该编码的求值需要 $\Theta((T_{\mathcal{M}}(|i|) + 1) \cdot |i| \cdot \log |i|)$ β-步，而参考编码只需 $\Theta(T_{\mathcal{M}}(|i|))$，因为每次转移都会读取输入。不过，时间复杂度仍然是多项式级别，因此就时间而言（以 β-步的数量来衡量），这种编码是合理的。

内在与数学带表示 图灵机带是一个字符串加上一个表示读写头的指定位置。van Emde Boas 在 [vEB12], 中将两种带表示分别称为内在表示和数学表示，介绍如下。

- 内在表示将字符串 i 和读写头的当前位置都表示为三元组 $i = i_l \cdot h \cdot i_r$，其中 i_l 和 i_r 分别是 i 中读写头所读取字符 h 两侧的前缀和后缀。在 λ 演算中对图灵机进行编码时，参考编码采用带的内在表示；如前所述，读取的时间开销为 $\mathcal{O}(1)$，但读取机制会带来 $\mathcal{O}(|i|)$ 的空间开销。

- 相反，磁带的数学表示只需给出 $n \in \mathbb{N}$ 读写头位置的索引，也就是说，三元组 $i_l \cdot h \cdot i_r$ 被替换为二元组（i，$|i_l| + 1$）。索引 $|i_l| + 1$ 的作用是指针；以二进制表示时，其大小为对数级。

  基于数学表示的 λ-演算编码，读取所需时间为 $\mathcal{O}$（$|i|\log$ $|i|$），空间开销为 $\mathcal{O}$（$\log$ $|i|$）。时间开销源于磁带的访问方式是顺序的：每次在磁带上移动一个单元格，并在每一步将索引减一，直到索引为 0。因此，访问右侧的单元格需要将索引减一 $|i_l| + 1$ 次，每次耗时 $\log$ $|i_l| + 1$，因为索引是以二进制表示的。

新的对数敏感编码为工作带保留内在表示，而为输入带采用数学表示。这是因为工作带的线性空间开销不会影响对数敏感性，而数学表示会使得在磁带上写入变得更加困难。

二进制算术与文献。为了在 λ-演算中实现读写头索引的操作，需要开发一种二进制串编码以及三种操作：后继、前驱和查找从 n 的二进制表示开始的字符串中的第 n 个字符。我们使用二进制字符串的 Scott 编码来实现这些操作，但采用反向表示二进制串。

我们从头开始，以一种看似临时拼凑的方式设计了二进制算术的编码。后来我们发现，我们的方法是文献中已有编码的一种变体。具体而言，它与 Mogensen 编码[非常相似，而 Mogensen 编码本身建立在 Goldberg 的研究 [之上01], 00]。

2 确定性 λ-演算与字符串的 Scott 编码

与参考编码的情况一样，新编码的像也是一种受限形式的 λ-演算，我们称之为确定性 λ-演算；在这种演算中，按名调用和按值调用求值恰好一致。

确定性 λ-演算。确定性 λ-演算的语言和求值上下文 $\Lambda_{\mathtt{det}}$ 定义如下：

$$\begin{array}{rrcl} \text{Terms} & t,u,r,w & ::= & v \mid tv \\ \text{Values} & v,w,v' & ::= & \lambda x.t \mid x \\ \\ \text{Evaluation Contexts} & E & ::= & \langle\cdot\rangle \mid Ev \end{array}$$

注意，

- 参数都是值：应用的右子项必须是一个值，这与普通 λ-演算中的情况不同。
- 弱求值：求值上下文是弱的，即不会进入抽象内部。

求值定义如下：

$$\begin{array}{cc} \textsc{Rule at top level} & \textsc{Contextual closure} \\ (\lambda x.t)u \mapsto_\beta t\{x\leftarrow u\} & E\langle t\rangle \to_{det} E\langle u\rangle \quad \text{if } t \mapsto_\beta u \end{array}$$

约定：为提高可读性，我们省略了一些括号，并规定应用的优先级高于抽象。因此，$\lambda x.tu$ 表示$\lambda x.(tu)$，而不是$(\lambda x.t)u$；后者需要用括号括起来。

这个演算的名称源于下面这个显然的引理。

引理 2.1（确定性）。设 $t \in \Lambda_{\mathtt{det}}$。若 $t \to_{det} u$ 且 $t \to_{det} r$，则 $u = r$。

证明。对 t作归纳。若 t 是变量或抽象，则它不能归约。若 $t = wv$，则分两种情况：

- w 是一个抽象 $\lambda x.s$。那么，$t = (\lambda x.s)v \to_{det} s\{x\leftarrow v\}$ 是 t唯一的可约表达式，即 $u = r = s\{x\leftarrow v\}$。
- w 不是抽象。那么，从 t 出发的两步归约分别来自两步归约$w \to_{det} u'$ 和 $w \to_{det} r'$，其中 $u = u'v$ 和 $r = r'v$，因为 $\langle\cdot\rangle v$ 是唯一可能的求值上下文。根据归纳假设，$u' = r'$，即 $u = r$。 □

不动点。图灵机的编码需要一般形式的递归，通常通过不动点组合子实现。我们使用图灵的不动点组合子，即其按值调用变体，它适用于 $\Lambda_{\mathtt{det}}$，并且返回一个在 η-等价意义下的不动点。令 fix 为项 $\theta\theta$，其中

$$\theta := \lambda x.\lambda y.y(\lambda z.xxyz).$$

现在，给定项 u，我们来证明 $\mathsf{fix}\, u$ 是 u 在 η-等价意义下的不动点。

$$\begin{array}{rcl} \mathsf{fix}\, u & = & (\lambda x.\lambda y.y(\lambda z.xxyz))\theta u \\ & \to_{det} & (\lambda y.y(\lambda z.\theta\theta yz))u \\ & \to_{det} & u(\lambda z.\theta\theta uz) \\ & = & u(\lambda z.\mathsf{fix}\, uz) \\ & =_{\eta} & u(\mathsf{fix}\, u) \end{array}$$

众所周知，在 λ-演算中，η-等价项无法区分（这就是 B¨ohm 定理）。因此，我们只需利用 $\mathsf{fix}\, u \to^2_{det} u(\lambda z.\mathsf{fix}\, uz)$ 这一事实，而无须处理 η-等价。这个事实不会带来任何复杂性。

字母表编码。设 $\Sigma = \{a_1, \ldots, a_n\}$ 是一个有限字母表。Σ 中的元素编码如下：

$$\ulcorner a_i \urcorner^{\Sigma} := \lambda x_1 . \ldots . \lambda x_n . x_i \ .$$

当上下文足以明确字母表时，我们只写 $\ulcorner a \urcorner_i$。注意，

1. 该表示为 Σ 确定了一个全序，使得 $a_i < a_j$ 当且仅当 $i < j$；
2. 元素 $\ulcorner a_i \urcorner^{\Sigma}$ 的表示所需空间与 $|\Sigma|$ 呈线性关系（而非对数关系）。但是，由于 Σ 是固定的，实际上所需空间为常数。

字符串编码。$s \in \Sigma^*$ 中的字符串由项 $\overline{s}^{\Sigma^*}$ 表示。我们的编码利用了这样一个事实：字符串是由若干字符连接而成，末尾跟着空字符串 ε（通常省略不写）。为此，编码使用 $|\Sigma| + 1$ 个抽象，其中额外的一个（下文定义中的 x_ε）用于表示 ε。编码按 s 的结构归纳定义如下：

$$\begin{array}{rcl} \overline{\varepsilon}^{\Sigma^*} & := & \lambda x_1 . \ldots \lambda x_n . \lambda x_\varepsilon . x_\varepsilon \ , \\ \overline{a_i r}^{\Sigma^*} & := & \lambda x_1 . \ldots \lambda x_n . \lambda x_\varepsilon . x_i \overline{r}^{\Sigma^*} . \end{array}$$

请注意，这种表示方式取决于 Σ的基数。不过，与之前一样，字母表是一个固定参数，因此这种依赖无关紧要。举例来说，字符串 aba 相对于按 $a < b$ 排序的字母表$\{a, b\}$ 的编码为

$$\overline{aba}^{\{a,b\}} = \lambda x_a . \lambda x_b . \lambda x_\varepsilon . x_a (\lambda x_a . \lambda x_b . \lambda x_\varepsilon . x_b (\lambda x_a . \lambda x_b . \lambda x_\varepsilon . x_a (\lambda x_a . \lambda x_b . \lambda x_\varepsilon . x_\varepsilon)))$$

引理 2.2（常数时间追加字符）。设 Σ 是一个字母表，$a \in \Sigma$ 是其中的一个字符。存在一个项 append^a_Σ，使得对于任意延续项 k 和任意字符串 $s \in \Sigma^*$,

$$\mathtt{append}^a_\Sigma k \overline{s} \to^{\mathcal{O}(1)}_{det} k\overline{(as)}.$$

证明。定义术语 $\mathsf{append}^a_\Sigma := \lambda k'.\lambda s'.k'(\lambda x_1 . \ldots . \lambda x_{|\Sigma|} . \lambda x_\varepsilon . x_{i_a} s')$，其中 i_a是 a 在由其编码确定的 Σ 顺序中的索引；该术语用于相对于字母表 Σ将字符 a 追加到字符串 s'。我们有：

$$\begin{array}{rcl}\mathtt{append}^a_\Sigma k\overline{s} & = & (\lambda k'.\lambda s'.k'(\lambda x_1.\ldots.\lambda x_{|\Sigma|}.\lambda x_\varepsilon.x_{i_a}s'))k\overline{s} \\ & \to^2_{det} & k(\lambda x_1.\ldots.\lambda x_{|\Sigma|}.\lambda x_\varepsilon.x_{i_a}\overline{s}) \\ & = & k\overline{(as)}.\end{array}$$

□

2.1 二进制算术

为了遍历输入词，我们考虑一个（二进制）计数器。将读写头向左（分别向右）移动，相当于将计数器减一（分别加一）。最初的想法是将一个数视为其二进制字符串表示，并使用字符串的 Scott 编码。由于在这种编码上定义后继和前驱函数颇为棘手，我们实际上定义了一种特设编码。

我们的编码有一个不寻常之处：二进制字符串是以逆序表示的，因此 2 的表示是 01，而不是 10。这样做是为了便于将后继函数和前驱函数定义为 λ-项；这些函数必须从左向右处理字符串，而采用标准表示时，它们必须先处理到字符串末尾，然后可能还要再从右向左处理。采用逆序表示后，这些函数只需从左向右处理字符串一次。

第二个不寻常之处是，为了避免由全为 0 的字符串以及右侧有许多 0 的字符串（这类字符串没有意义）引起的问题，我们将所有由 0 构成的后缀都折叠为空字符串。因此，数字 0 就用空字符串表示。非最右侧的 0 位则采用通常的 Scott 编码表示。

如果 $n \in \mathbb{N}$ 我们用 $\lfloor n \rfloor$ 表示代表 n的二进制字符串，那么有：

$$\begin{array}{rcl}\lfloor 0 \rfloor & := & \varepsilon \\ \lfloor 1 \rfloor & := & 1 \\ \lfloor 2 \rfloor & := & 01 \\ \lfloor 3 \rfloor & := & 11 \\ \lfloor 4 \rfloor & := & 001\end{array}$$

依此类推。随后，使用 Scott 编码将二进制串编码为 λ-项，如下所示：

$$\begin{array}{rcl}\overline{\varepsilon} & := & \lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_\varepsilon \\ \overline{0{\cdot}s} & := & \lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_0\overline{s} \\ \overline{1{\cdot}s} & := & \lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_1\overline{s}\end{array}$$

后继函数。在反转的二进制表示上，后继函数 $\textsc{succ}$ 可以定义如下（使用类 Haskell 语法）：

$$\begin{array}{lcl}\textsc{succ}\ \ \varepsilon & = & 1 \\ \textsc{succ}\ \ 0{\cdot}s & = & 1{\cdot}s \\ \textsc{succ}\ \ 1{\cdot}s & = & 0{\cdot}(\textsc{succ}\ s)\end{array}$$

对于该项，我们有 $\textsc{succ}(\lfloor n \rfloor) = \lfloor n+1 \rfloor$。

引理 2.3。存在一个 λ-项 succ，使得对于每个延续项 k以及每个自然数 $n \in \mathbb{N}$,

$$\mathsf{succ}\, k\overline{\lfloor n \rfloor} \to_{det}^{\mathcal{O}(\log n)} k\,\overline{\mathrm{SUCC}\,\lfloor n \rfloor}.$$

证明。定义 succ := Θsuccaux 和 succaux := $\lambda f.\lambda k'.\lambda n'.n' N_0 N_1 N_\varepsilon f k'$如下：

- $N_0 := \lambda f'.\lambda s'.\lambda k'.$追加$^1 k' s'$
- $N_1 := \lambda f'.\lambda s'.\lambda k'.f'($$\lambda z.$追加$^0 k' z)s'$
- $N_\varepsilon := \lambda f'.\lambda k'.k'\,\overline{1{\cdot}\varepsilon}$

我们改为证明 $\mathsf{succ}\, k\overline{\lfloor n \rfloor} \to_{det}^{\mathcal{O}(|\lfloor n \rfloor|)} k\,\overline{\mathrm{SUCC}\,\lfloor n \rfloor}$.，显然 $|\lfloor n \rfloor| = \log\, n$，因为自然可以通过对 $\lfloor n \rfloor$ 作为字符串的长度进行归纳法来证明。求值 $\mathsf{succ}\, k\overline{\lfloor n \rfloor}$ 的前几步对所有自然数$n \in \mathbb{N}$都相同：

$$\begin{array}{rcl}
\mathsf{succ}\, k\overline{\lfloor n \rfloor} & = & \mathsf{fix}\,\mathsf{succaux}\, k\overline{\lfloor n \rfloor} \\
& \to_\beta^2 & \mathsf{succaux}(\lambda z.\mathsf{succ}\, z)k\overline{\lfloor n \rfloor} \\
& = & (\lambda f.\lambda k'.\lambda n'.n' N_0 N_1 N_\varepsilon f k')(\lambda z.\mathsf{succ}\, z)k\overline{\lfloor n \rfloor} \\
& \to_\beta^3 & \overline{\lfloor n \rfloor} N_0 N_1 N_\varepsilon (\lambda z.\mathsf{succ}\, z)k
\end{array}$$

n的情况：

- 零，也就是 $n = 0$、$\lfloor n \rfloor = \varepsilon$ 和 $\overline{\lfloor n \rfloor} = \lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_\varepsilon$：那么

$$\begin{array}{rcl}
\overline{\lfloor n \rfloor} N_0 N_1 N_\varepsilon (\lambda z.\mathsf{succ}\, z)k & = & (\lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_\varepsilon)N_0 N_1 N_\varepsilon (\lambda z.\mathsf{succ}\, z)k \\
& \to_\beta^3 & N_\varepsilon (\lambda z.\mathsf{succ}\, z)k \\
& = & (\lambda f'.\lambda k'.k'\,\overline{1{\cdot}\varepsilon})k \\
& \to_\beta & k\,\overline{1{\cdot}\varepsilon} \\
& = & k\,\overline{\mathrm{SUCC}\,\lfloor 0 \rfloor}
\end{array}$$

- 非零。因此，根据字符串 $\lfloor n \rfloor$的首字符，可分为两种子情况：

  - 0 *character*, i.e. $\lfloor n \rfloor = 0{\cdot}s$: then

$$\begin{array}{rcll}
 & & \overline{0{\cdot}r} N_0 N_1 N_\varepsilon (\lambda z.\mathsf{succ}\, z)k & \\
 & = & (\lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_0 \overline{s})N_0 N_1 N_\varepsilon (\lambda z.\mathsf{succ}\, z)k & \\
 & \to_\beta^3 & N_0 \overline{s}(\lambda z.\mathsf{succ}\, z)k & \\
 & = & (\lambda f'.\lambda s'.\lambda k'.\mathsf{append}^1 k' s')\overline{s}(\lambda z.\mathsf{succ}\, z)k & \\
 & \to_\beta^3 & \mathsf{append}^1 k \overline{s} & \\
(L.\ 2.2) & \to_\beta^{\mathcal{O}(1)} & k\,\overline{1{\cdot}s} & \\
 & = & k\,\overline{\mathrm{SUCC}\,\lfloor n \rfloor} &
\end{array}$$

– 1 个字符，即 $\lfloor n \rfloor = 1{\cdot}s$：那么

$$\begin{array}{rll}
 & & \overline{1{\cdot}r} N_0 N_1 N_\varepsilon (\lambda z.\mathtt{succ}\, z) k \\
 & = & (\lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_1 \overline{s}) N_0 N_1 N_\varepsilon (\lambda z.\mathtt{succ}\, z) k \\
 & \to_\beta^3 & N_1 \overline{s} (\lambda z.\mathtt{succ}\, z) k \\
 & = & (\lambda f'.\lambda s'.\lambda k'.f'(\lambda z.\mathtt{append}^0 k' z) s') \overline{s} (\lambda z.\mathtt{succ}\, z) k \\
 & \to_\beta^3 & (\lambda z.\mathtt{succ}\, z)\, (\lambda z.\mathtt{append}^0 k z) \overline{s} \\
 & \to_\beta & \mathtt{succ}\, (\lambda z.\mathtt{append}^0 k z) \overline{s} \\
(i.h.) & \to_\beta^{\mathcal{O}(|s|)} & (\lambda z.\mathtt{append}^0 k z) \overline{\mathrm{SUCC}\, s} \\
 & \to_{det} & \mathtt{append}^0 k \overline{\mathrm{SUCC}\, s} \\
(L.\ 2.2) & \to_\beta^{\mathcal{O}(1)} & k\, \overline{0{\cdot}(\mathrm{SUCC}\, s)} \\
 & = & k\, \overline{\mathrm{SUCC}\, (1{\cdot}s)} \\
 & = & k\, \overline{\mathrm{SUCC}\, \lfloor n \rfloor}
\end{array}$$

□

前驱函数。现在我们定义并实现一个前驱函数。我们假定它只会作用于不同于 0 的自然数 n 的编码 $\lfloor n \rfloor$，因为下文确实如此。这样的前驱函数 PRED 在反向二进制表示上的定义如下（使用类 Haskell 语法）：

$$\begin{array}{llcl}
\mathrm{PRED} & 0{\cdot}s & = & 1{\cdot}(\mathrm{PRED}\, s) \\
\mathrm{PRED} & 1{\cdot}\varepsilon & = & \varepsilon \\
\mathrm{PRED} & 1{\cdot}b{\cdot}s & = & 0{\cdot}b{\cdot}s
\end{array}$$

容易看出，对于所有 $0 < n \in \mathbb{N}$，$\mathrm{PRED}(\lfloor n \rfloor) = \lfloor n-1 \rfloor$。请注意，当 $\mathrm{PRED}(\lfloor n \rfloor)$ 改变 $\lfloor n \rfloor$ 的最右位时，并不会引入一个最右侧的 0 位，也就是说，结果是 $\mathrm{PRED}\, 001 = 11$，而不是 110。

引理 2.4。存在一个 λ-项 pred，使得对于每个延续项 k 和每个自然数 $1 \leq n \in \mathbb{N}$，

$$\mathtt{pred}\, k \overline{\lfloor n \rfloor} \to_{det}^{\mathcal{O}(\log n)} k\, \overline{\mathrm{PRED}\, \lfloor n \rfloor}.$$

证明。定义 pred := fix predaux 和 predaux := $\lambda f.\lambda k'.\lambda n'.n' N_0 N_1 N_\varepsilon f k'$，其中：

- $N_0 := \lambda r'.\lambda f.\lambda k'.f(\lambda z.\mathtt{append}^1 k' z) r'$;
- $N_1 := \lambda r'.\lambda f.r' M_0 M_1 M_\varepsilon$, where:
  - $M_0 := \lambda v.\lambda k.\mathtt{append}^0 (\lambda z.\mathtt{append}^1 k z) v$;
  - $M_1 := \lambda v.\lambda k.\mathtt{append}^1 (\lambda z.\mathtt{append}^1 k z) v$;
  - $M_\varepsilon := \lambda k'.k' \overline{\varepsilon}$;
- N_ε 是任意闭项。

我们更倾向于证明前驱 $k\overline{\lfloor n \rfloor} \to_{det}^{\mathcal{O}(|\lfloor n \rfloor|)} k\, \overline{\mathrm{PRED}\, \lfloor n \rfloor}$.，显然 $|\lfloor n \rfloor| = \log\, n$，因为该证明自然是对 $\lfloor n \rfloor$ 作为字符串的长度进行归纳法。该

$\mathsf{pred}\,k\overline{\lfloor n\rfloor}$ 的求值的最初步骤对所有自然数$1 \le n \in \mathbb{N}$都是相同的：

$$\begin{array}{rcl}
\mathtt{pred}\,k\overline{\lfloor n\rfloor} & = & \mathsf{fix}\,\mathtt{predaux}\,k\overline{\lfloor n\rfloor} \\
& \to_\beta^2 & \mathtt{predaux}(\lambda z.\mathtt{pred}\,z)k\overline{\lfloor n\rfloor} \\
& = & (\lambda f.\lambda k'.\lambda n'.n'N_0N_1N_\varepsilon fk')(\lambda z.\mathtt{pred}\,z)k\overline{\lfloor n\rfloor} \\
& \to_\beta^3 & \overline{\lfloor n\rfloor}N_0N_1N_\varepsilon(\lambda z.\mathtt{pred}\,z)k
\end{array}$$

根据假设，$n \ge 1$。那么$\overline{n}$是一个非空字符串。其首字符的情况如下：

- 0 字符，即 $\lfloor n\rfloor = 0{\cdot}r$：那么

$$\begin{array}{rcl}
 & & \overline{\lfloor n\rfloor}N_0N_1N_\varepsilon(\lambda z.\mathtt{pred}\,z)k \\
 & = & (\lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_0\overline{r})N_0N_1N_\varepsilon(\lambda z.\mathtt{pred}\,z)k \\
 & \to_\beta^3 & N_0\overline{r}(\lambda z.\mathtt{pred}\,z)k \\
 & = & (\lambda r'.\lambda f.\lambda k'.f(\lambda z.\mathtt{append}^1k'z)r')\overline{r}(\lambda z.\mathtt{pred}\,z)k \\
 & \to_\beta^3 & (\lambda z.\mathtt{pred}\,z)(\lambda z.\mathtt{append}^1k'z)\overline{r} \\
 & \to_\beta & \mathtt{pred}(\lambda z.\mathtt{append}^1kz)\overline{r} \\
(i.h.) & \to_\beta^{\mathcal{O}(|r|)} & (\lambda z.\mathtt{append}^1kz)\,\overline{\mathrm{PRED}\,r} \\
 & \to_\beta & \mathtt{append}^1k\,\overline{\mathrm{PRED}\,r} \\
(L.\ 2.2) & \to_\beta^{\mathcal{O}(1)} & k\,\overline{1{\cdot}(\mathrm{PRED}\,r)} \\
 & = & k\,\overline{\mathrm{PRED}\,0{\cdot}r} \\
 & = & k\,\overline{\mathrm{PRED}\,\lfloor n\rfloor}
\end{array}$$

- 1 字符，即 $\lfloor n\rfloor = 1{\cdot}r$：那么

$$\begin{array}{rcl}
\overline{\lfloor n\rfloor}N_0N_1N_\varepsilon(\lambda z.\mathtt{pred}\,z)k & = & (\lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_1\overline{r})N_0N_1N_\varepsilon(\lambda z.\mathtt{pred}\,z)k \\
 & \to_\beta^3 & N_1\overline{r}(\lambda z.\mathtt{pred}\,z)k \\
 & = & (\lambda r'.\lambda f.r'M_0M_1M_\varepsilon)\overline{r}(\lambda z.\mathtt{pred}\,z)k \\
 & \to_\beta & \overline{r}M_0M_1M_\varepsilon k
\end{array}$$

根据字符串 r，有三种子情况：

- r 为空，即 $r = \varepsilon$。那么：

$$\begin{array}{rcl}
\overline{\varepsilon}M_0M_1M_\varepsilon k & = & (\lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_\varepsilon)M_0M_1M_\varepsilon k \\
 & \to_\beta^3 & M_\varepsilon k \\
 & = & (\lambda k'.k'\overline{\varepsilon})k \\
 & \to_\beta & k\overline{\varepsilon} \\
 & \to_\beta & k\overline{\lfloor 0\rfloor} \\
 & = & k\overline{\mathrm{PRED}\,\lfloor 1\rfloor}
\end{array}$$

– r 以 **0** 开头，也就是说，$\lfloor n \rfloor = 1{\cdot}r = 1{\cdot}0{\cdot}p_\circ$。那么：

$$\begin{array}{rcl}
\overline{0{\cdot}p}\, M_0 M_1 M_\varepsilon k & = & (\lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_0\overline{p})M_0M_1M_\varepsilon k \\
& \to_\beta^3 & M_0\overline{p}k \\
& = & (\lambda v.\lambda k.\mathtt{append}^0(\lambda z.\mathtt{append}^0 kz)v)\overline{p}k \\
& \to_\beta^2 & \mathtt{append}^0(\lambda z.\mathtt{append}^0 kz)\overline{p} \\
(L.\ 2.2) & \to_\beta^{\mathcal{O}(1)} & (\lambda z.\mathtt{append}^0 kz)\overline{0{\cdot}p} \\
& \to_\beta & \mathtt{append}^0 k\,\overline{0{\cdot}p} \\
(L.\ 2.2) & \to_\beta^{\mathcal{O}(1)} & k\,\overline{0{\cdot}0{\cdot}p} \\
& = & k\,\overline{0{\cdot}r} \\
& = & k\,\overline{\mathrm{PRED}\ 1{\cdot}r} \\
& = & k\,\overline{\mathrm{PRED}\ \lfloor n \rfloor}
\end{array}$$

– r 从 **1** 开始，即 $\lfloor n \rfloor = 1{\cdot}r = 1{\cdot}1{\cdot}p_\circ$。然后：

$$\begin{array}{rcl}
\overline{0{\cdot}p}\, M_0 M_1 M_\varepsilon k & = & (\lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_1\overline{p})M_0M_1M_\varepsilon k \\
& \to_\beta^3 & M_1\overline{p}k \\
& = & (\lambda v.\lambda k.\mathtt{append}^1(\lambda z.\mathtt{append}^0 kz)v)\overline{p}k \\
& \to_\beta^2 & \mathtt{append}^1(\lambda z.\mathtt{append}^0 kz)\overline{p} \\
(L.\ 2.2) & \to_\beta^{\mathcal{O}(1)} & (\lambda z.\mathtt{append}^0 kz)\overline{1{\cdot}p} \\
& \to_\beta & \mathtt{append}^0 k\,\overline{1{\cdot}p} \\
(L.\ 2.2) & \to_\beta^{\mathcal{O}(1)} & k\,\overline{0{\cdot}1{\cdot}p} \\
& = & k\,\overline{0{\cdot}r} \\
& = & k\,\overline{\mathrm{PRED}\ 1{\cdot}r} \\
& = & k\,\overline{\mathrm{PRED}\ \lfloor n \rfloor}
\end{array}$$

□

查找函数。给定一个自然数 n，我们需要能够从非空字符串 s 中提取第 $n+1$ 个字符。偏函数 LOOKUP可以如下定义（采用类 Haskell 语法）：

$$\begin{array}{llcl}
\mathrm{LOOKUP} & \lfloor 0 \rfloor \quad (c{\cdot}s) & = & c \\
\mathrm{LOOKUP} & \lfloor n \rfloor \quad (c{\cdot}s) & = & \mathrm{LOOKUP}\ (\mathrm{PRED}\ \lfloor n \rfloor)\ s \quad \text{if } n > 0
\end{array}$$

引理 2.5。存在一个 λ-项 lookup，使得对于任意延续项k、任意自然数 n 以及任意非空字符串 $i \in \mathbb{B}^+$，

$$\mathtt{lookup}\, k\overline{\lfloor n \rfloor i} \quad \to_{det}^{\mathcal{O}(n \log n)} \quad k\,\lceil \mathrm{LOOKUP}\ \lfloor n \rfloor i \rceil.$$

证明。现在我们可以将函数 lookup 编码为：$=$ fixlookupaux，其中：

$$\mathtt{lookupaux} := \lambda f.\lambda k'.\lambda n'.\lambda i'.n' N_0 N_1 N_\varepsilon f k' i'$$

其中：

- $N_0 := \lambda p'.\lambda f.\lambda k'.\lambda i'.i' M_0 M_0 M_\varepsilon p' f k'$, where
  - $M_0 := \lambda r'.\lambda p'.\lambda f.\lambda k'.\mathtt{append}^0(\lambda z''.\mathtt{pred}(\lambda z'.fk'z')z'')p'r'$;
  - M_ε is whatever closed term.

- $N_1 := \lambda p'.\lambda f.\lambda k'.\lambda i'.i' M_1 M_1 M_\varepsilon p' f k'$, where
  - $M_1 := \lambda r'.\lambda p'.\lambda f.\lambda k'.\mathtt{append}^1(\lambda z''.\mathtt{pred}(\lambda z'.fk'z')z'')p'r'$;
  - M_ε is whatever closed term.
- $N_\varepsilon := \lambda f.\lambda k'.\lambda i'.i' O_0 O_1 O_{\mathsf{L}} O_{\mathsf{R}} O_\varepsilon k'$, where
  - $O_b := \lambda s'.\lambda k'.k'\lceil b\rceil$;
  - O_ε is whatever closed term.

$\mathsf{lookup}\, k\overline{\lfloor n\rfloor}\overline{i}$ 求值的最初步骤对所有字符串 $i \in \mathbb{B}^+$和自然数 $n \in \mathbb{N}$都是相同的：

$$\begin{array}{lll} \mathtt{lookup}\, k\overline{\lfloor n\rfloor}\overline{i} & = & \mathsf{fix}\, \mathtt{lookupaux}\, k\overline{\lfloor n\rfloor}\overline{i} \\ & \to_\beta^2 & \mathtt{lookupaux}(\lambda z.\mathtt{lookup}\, z)k\overline{\lfloor n\rfloor}\overline{i} \\ & = & (\lambda f.\lambda k'.\lambda n'.\lambda i'.n' N_0 N_1 N_\varepsilon f k' i')(\lambda z.\mathtt{lookup}\, z)k\overline{\lfloor n\rfloor}\overline{i} \\ & \to_\beta^4 & \overline{\lfloor n\rfloor} N_0 N_1 N_\varepsilon (\lambda z.\mathtt{lookup}\, z)k\overline{i} \end{array}$$

n的情况：

- $n = 0$, and so $\lfloor n\rfloor = \varepsilon$: then

$$\begin{array}{ll} & \overline{\varepsilon} N_0 N_1 N_\varepsilon (\lambda z.\mathtt{lookup}\, z)k\overline{i} \\ = & (\lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_\varepsilon) N_0 N_1 N_\varepsilon (\lambda z.\mathtt{lookup}\, z)k\overline{i} x_\varepsilon N_\varepsilon (\lambda z.\mathtt{lookup}\, z)k\overline{i} \\ = & (\lambda f.\lambda k'.\lambda i'.i' O_0 O_1 O_{\mathsf{L}} O_{\mathsf{R}} O_\varepsilon k')(\lambda z.\mathtt{lookup}\, z)k\overline{i} \\ \to_{det}^3 & \overline{i} O_0 O_1 O_{\mathsf{L}} O_{\mathsf{R}} O_\varepsilon k \end{array}$$

  设 i 以 $b \in \mathbb{B}$ 开头，即 $i = b{\cdot}s$:

$$\begin{array}{lll} \overline{b{\cdot}s}\, O_0 O_1 O_{\mathsf{L}} O_{\mathsf{R}} O_\varepsilon k & = & (\lambda x_0.\lambda x_1.\lambda x_{\cdot\varepsilon} x_b \overline{s}) O_0 O_1 O_{\mathsf{L}} O_{\mathsf{R}} O_\varepsilon k \\ & \to_{det}^3 & O_b \overline{s} k \\ & = & (\lambda s'.\lambda k'.k'\lceil b\rceil)\overline{s}k \\ & \to_\beta^2 & k\lceil b\rceil \\ & = & k\lceil \textsc{lookup}\, \varepsilon(b{\cdot}s)\rceil \\ & = & k\lceil \textsc{lookup}\, \lfloor 0\rfloor i\rceil \end{array}$$

- 以 0 开头的非空字符串，即 $n > 0$ 和 $\lfloor n\rfloor = 0{\cdot}p$：那么

$$\begin{array}{ll} & \overline{\lfloor n\rfloor} N_0 N_1 N_\varepsilon (\lambda z.\mathtt{lookup}\, z)k\overline{i} \\ = & (\lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_0 \overline{p}) N_0 N_1 N_\varepsilon (\lambda z.\mathtt{lookup}\, z)k\overline{i} \\ \to_\beta^3 & N_0 \overline{p} (\lambda z.\mathtt{lookup}\, z)k\overline{i} \\ = & (\lambda p'.\lambda f.\lambda k'.\lambda i'.i' M_0 M_0 M_\varepsilon p' f k')\overline{p}(\lambda z.\mathtt{lookup}\, z)k\overline{i} \\ \to_\beta^4 & \overline{i} M_0 M_0 M_\varepsilon \overline{p} (\lambda z.\mathtt{lookup}\, z)k \end{array}$$

  令 i 以 $b \in \mathbb{B}$ 开始，也就是说， $i = b{\cdot}r$:

$\overline{b\cdot r}M_0M_0M_\varepsilon k =_{\beta}(\lambda x_0.\lambda x_1.\lambda x_\varepsilon.x_b\overline{r})M_0M_0M_\varepsilon\overline{p}(\lambda z.\text{查找 }z)k \to_{det}^{|\Sigma|} M_0\overline{r}\,\overline{p}(\lambda z.\text{查找 }z)k =_{\beta}(\lambda r'.\lambda p'.\lambda f.\lambda k'.\text{追加}^0(\lambda z''.\text{前驱}(\lambda z'.fk'z')z'')p'r')\overline{r}\,\overline{p}(\lambda z.\text{查找 }z)k \to_{det}^{4}\text{追加}^0(\lambda z''.\text{前驱}(\lambda z'.(\lambda z.\text{查找 }z)kz')z'')\overline{p}\,\overline{r}(L.\ 2.2 \to_{det}^{\mathcal{O}(1)}(\lambda z''.\text{前驱}(\lambda z'.(\lambda z.\text{查找 }z)kz')z'')\overline{0\cdot p}\,\overline{r} \to_{det}\text{前驱}(\lambda z'.(\lambda z.\text{查找 }z)kz')\overline{0\cdot p}\,\overline{r} =_{\beta}\text{前驱}(\lambda z'.(\lambda z.\text{查找 }z)kz')\overline{\lfloor n\rfloor}\,\overline{r}(L.\ 2.4)\ \to_{det}^{\mathcal{O}(\log n)}(\lambda z'.(\lambda z.\text{查找 }z)kz')\overline{\text{PRED}\,\lfloor n\rfloor}\,\overline{r} \to_{det}^{2}\text{查找}\,k\,\overline{\text{PRED}\,\lfloor n\rfloor}\,\overline{r} =_{\beta}\text{查找 }k\,\overline{\lfloor n-1\rfloor}\,\overline{r}(\text{即})\ \to_{det}^{\mathcal{O}((n-1)\cdot\log(n-1))}k(\lceil\text{LOOKUP}\,\lfloor n-1\rfloor r\rceil) =_{\beta} k(\lceil\text{LOOKUP}\,\lfloor n\rfloor s\rceil)$

因此， β 步数为 $\mathcal{O}(\log\ n)\ +\mathcal{O}((n-1)\ \cdot\log(n-1))\ +h$，其中 h 是某个常数；该步数有上界 $\mathcal{O}(n\cdot\log n)$，符合要求。

- 以 1 开头的非空字符串，即 $n>0$ 和 $\lfloor n\rfloor=1{\cdot}p$：与前一个相同，只需将 N_0 替换为 N_1，因此也将 M_0 替换为 M_1。特别地，它所需的步数相同。

□

2.2 图灵机的新编码

图灵机。令 $\mathbb{B}_{\mathsf{I}}:=\{0,1,\mathrm{L},\mathrm{R}\}$ 和 $\mathbb{B}_{\mathsf{W}}:=\{0,1,\square\}$，其中 L 和 R 界定输入（二进制）字符串，而 □ 是我们对空白符号的记法。带输入的确定性二进制图灵机 $\mathcal{M}$ 是一个由 $(Q,q_{in},q_T,q_F,\delta)$ 组成的元组，其中包括：

- 一个状态的有限集合 $Q=\{q_1,\ldots,q_m\}$；
- 一个特殊状态 $q_{in}\in Q$，称为初始状态；
- 两个特殊状态 $Q_{fin}:=\{q_T,q_F\}\subseteq Q$，称为终止状态；
- 一个部分转移函数 $\delta\colon\ \mathbb{B}_{\mathsf{I}}\times\mathbb{B}_{\mathsf{W}}\times Q\rightharpoonup\{-1,+1,0\}\times\mathbb{B}_{\mathsf{W}}\times\{\leftarrow,\rightarrow,\downarrow\}\times Q$ 满足：仅当 $q\notin Q_{fin}$ 时，才定义 $\delta(b,a,q)$。

$\mathcal{M}$ 一个配置是一个元组

$$(i, n, w_l, a, w_r, q) \in \mathbb{B}_{\mathsf{I}}^* \times \mathbb{N} \times \mathbb{B}_{\mathsf{W}}^* \times \mathbb{B}_{\mathsf{W}} \times \mathbb{B}_{\mathsf{W}}^* \times Q$$

其中：

- i 是不可变输入串，形式为 $i = \mathrm{L}{\cdot}s{\cdot}\mathrm{R} \quad s \in \mathbb{B}^*$；
- $n \in \mathbb{N}$ 表示输入头的位置。它应以二进制表示（即作为 $\mathbb{B}^*$ 的一个元素），以使所占空间为 $\log n$；但为便于阅读，我们仍将其称为数字，而非字符串；
- ${w_l} \in {\mathbb{B}^*}_{\mathsf{W}}$ 是工作头左侧的工作带；
- $a \in \mathbb{B}_{\mathsf{W}}$ 是工作头读取的工作带单元格中的元素；
- ${w_r} \in {\mathbb{B}^*}_{\mathsf{W}}$ 是工作头右侧的工作带；
- $q \in Q$ 是机器的状态。

为了便于阅读，我们通常将配置 (i, n, w_l, a, w_r, q) 写作 $(i, n \mid w_l, a, w_r \mid q)$，以区分输入分量、工作分量和当前状态。

给定输入字符串 $i \in \mathbb{B}_{\mathsf{I}}^*$（其中 $i = \mathrm{L}{\cdot}s{\cdot}\mathrm{R}$ 且 $s \in \mathbb{B}^*$），我们定义：

- 对于 i，初始配置 $C_{\mathtt{in}}(i)$ 为 $C_{\mathtt{in}}(i) := (i, 0 \mid \varepsilon, \square, \varepsilon \mid q_{in})$，
- 最终配置 $C_{\mathtt{fin}} := (s, n \mid w_l, a, w_r \mid q)$，其中 $q \in Q_{fin}$。

为了便于阅读，转移（例如 $\delta(i_n, a, q) = (-1, a', \leftarrow, q')$）通常写作 $(-1 \mid a', \leftarrow \mid q')$，以突出它与配置（输入、工作、状态）对应的三个分量。

与 Goldreich 一样，我们假设机器不会扫描输入边界之外的内容。这不会影响空间复杂度。

转移示例：若 $\delta(i_n, a, q) = (-1 \mid a', \leftarrow \mid q')$，则 $\mathcal{M}$ 从 $C = (i, n \mid w_l a'', a, w_r \mid q)$ 演变为 $D = (i, n-1 \mid w_l, a'', a' w_r \mid q')$；其中，$i$ 的第 n 个字符是 i_n。若读写头左侧的磁带为空，即 $C = (i, n \mid \varepsilon, a, w_r \mid q)$，则新读写头单元格的内容为空白符号，也就是说，$D := (i, n-1 \mid \varepsilon, \square, a' w_r \mid q')$。读写头右侧的磁带为空时，情况也相同。若 $\mathcal{M}$ 有一条从 C 到 D 的转移，我们记作 $C \to_{\mathcal{M}} D$。状态为最终状态 $q \in Q_{fin}$ 的配置是最终配置，不能继续演变。

图灵机 $(Q, q_{in}, q_T, q_F, \delta)$ 计算函数 $f\colon \mathbb{B}^* \to \mathbb{B}$ 在时间 $T\colon \mathbb{N} \to \mathbb{N}$ 和空间 $S\colon \mathbb{N} \to \mathbb{N}$ 内完成，当且仅当对于每个 $i \in \mathbb{B}^+$，i 的初始配置会在 $T(|i|)$ 步内演化为状态为 $q_{f(i)}$ 的最终配置，并且工作带上至多使用 $S(|i|)$ 个单元格。

构型编码。机器 $\mathcal{M} = (Q, q_{in}, q_T, q_F, \delta)$ 的配置 $(i, n \mid s, a, r \mid q)$ 用以下项表示

$$\overline{(i, n \mid w_l, a, w_r \mid q)}^{\mathcal{M}} := \lambda x.(x \overline{i}^{\mathbb{B}^+} \overline{\lfloor n \rfloor}^{\mathbb{B}} \overline{w_l^{\mathrm{R}}}^{\mathbb{B}_{\mathsf{W}}^*} \lceil a \rceil^{\mathbb{B}_{\mathsf{W}}} \overline{w_r}^{\mathbb{B}_{\mathsf{W}}^*} \lceil q \rceil^{Q}).$$

其中 w_l^{R} 是将元素顺序反转后的字符串 w_l。我们通常更倾向于写作

$$\overline{(i, n \mid w_l, a, w_r \mid q)} := \lambda x.(x\ \overline{i}\ \overline{\lfloor n \rfloor}\ \overline{w_l^{\mathrm{R}}}\ \lceil a \rceil\ \overline{w_r}\ \lceil q \rceil).$$

省略上标。为便于阅读，我们有时使用以下记法表示元组 $\langle s、\ q \mid t、\ u、\ r \mid w\rangle :=\lambda x.(xsqturw)$，因此 $(i、\ n \mid w_l、\ a、\ w_r \mid q) =\langle \overline{i}、\ \overline{\lfloor n \rfloor} \mid \overline{w_l^{\mathrm{R}}}、\ \lceil a \rceil、\ \overline{w_r} \mid \lceil q \rceil\rangle$。

转移函数的编码 转移函数 $\delta(b, a, q)$ 通过查找一个三维表 T 来实现，其坐标为：

- 输入：输入带上的当前位 b，它实际上是从输入带 i 和表示当前输入位置的计数器 n 中获取的，
- 工作：工作带上的当前字符 a，以及
- 状态：当前状态 q，

转移函数编码为一个递归的 λ-项 trans，它以 i、n——用于获取 b——以及 a 和 q的编码作为参数。其工作方式如下：

- 它首先通过应用查找函数，从 n 和 i 中获取 b；
- 它有一个子项 A_b，对应 b的四个值。通过将 $\lceil b \rceil$ 应用于 b 的编码以及 A_0、A_1、A_{L} 和 A_{R}，即可选出右侧子项。
- 每个 A_b 依次都有一个子项$B_{b,\ a}$，对应于每个字符 $a \in \mathbb{B}_{\mathsf{W}}$，即工作带的坐标。通过将 $\lceil a \rceil$ 应用于工作带上当前字符 a 的编码以及$B_{b,\ 0}$、$B_{b,\ 1}$、$B_{b,\ \square}$，即可选出右侧子项。
- 每个$B_{b,\ a}$ 依次都有一个子项$C_{b,\ a,\ q}$，对应 q 中的每个字符。通过将 $\lceil q \rceil$ 应用于当前状态q的编码以及$C_{b,\ a,\ q_1}$、.....、$B_{b,\ a,\ q_{|Q|}}$，即可选出右侧子项。
- 子项$C_{b,\ a,\ q}$ 根据转移函数 δ生成下一个配置（的编码）。如果 δ 使输入带的计数器减小（相应地，增大），那么$C_{b,\ a,\ q}$ 会对输入计数器应用前驱函数（相应地，后继函数），然后应用一个与工作带上所需操作相对应的项，即：
  - S（表示停留），即读写头不移动时的情况。这种情况很简单，S 只需生成下一个配置。– L 表示读写头向左移动。设 $w_l = wa''$ 为 a'，即转移需要写入的字符；设 q' 为新状态。那么 L 有一个子项 $L^{a'\ q'}_{\ a''}$，针对每个 $a'' \in \mathbb{B}_{\mathsf{W}}$，其作用是将 a' 添加到工作带右侧部分，从工作带左侧部分移除 a''（移除后变为 w），并将 a'' 设为读写头位置上的字符。– R 表示读写头向右移动。其结构与 L 类似。

  为了尽可能保持模块化，我们在输入头移动和不移动的情况下，都使用 S、L 和 R 的定义，即使这需要额外更新计数器 n，虽然这次更新没有用处，但无害。

将项 trans$^{\mathcal{M}}$（或简称 trans）定义如下。

$$\begin{array}{rcl} \underset{a}{\texttt{transaux}} & \underset{a}{:=} & \lambda x.\lambda k'.\lambda C'.C'(\lambda i'.\lambda n'.\lambda w'_l.\lambda a'.\lambda w'_r.\lambda q'.\texttt{lookup}\, K i' n') \\ \texttt{trans} & := & \mathsf{fix}\,\texttt{transaux}, \end{array}$$

其中:

$$\begin{array}{rcl} T & := & \lambda b'.b'\, A_0 A_1 A_{\mathsf{L}} A_{\mathsf{R}} a' q' x k' i' n' w'_l w'_r A_b \quad := \quad \lambda a'.a' B_{b,0} B_{b,1} B_{b,\square} B_b, \\ C_{b\ a} & := & \lambda q'.q' C_{b,\ ,q_1} \ldots C_{b,\ ,\ ,\ ,q} := \quad \lambda x.\lambda k'.\lambda i'.\lambda n'.\lambda w'_l.\lambda w'_r. \end{array} \left\{ \begin{array}{l} k'\langle i' \\ \\ Ln' \\ Rn' \\ \\ \\ \\ \\ \\ \end{array} \right. ,$$

$$\begin{array}{rcl} n' \mid w'_l\ \lceil a \rceil\ w'_r \mid \lceil q \rceil \rangle & & q \in Q_{fin}, , \text{ if if } (, , q)\ (0 \mid a', \downarrow \mid q') \text{ if } \delta(b,a,q) = (0 \mid a', \leftarrow \mid q') \text{ if } \delta(b,a, \\ Sn' & \delta\ b\ a & = \\ q) = (0 \mid a', \rightarrow \mid q') \text{ 前驱} Sn' \text{ if } \delta(b,a,q) = (-1 \mid a', \downarrow \mid q') \text{ 前驱} Ln' \text{ if } \delta(b,a,q) = (-1 \mid a', \leftarrow \mid q') \text{ 前驱} \\ Rn' \text{ if } \delta(b,a,q) = (-1 \mid a', \rightarrow \mid q') \text{ 后继函数} Sn' \text{ if } \delta(b,a,q) = (+1 \mid a', \downarrow \mid q') \text{ 后继函数} Ln' \text{ if } \delta(b,a,q) \\ = (+1 \mid a', \leftarrow \mid q') \text{ 后继函数} Rn' \text{ if } \delta(b,a,q) = (+1 \mid a', \rightarrow \mid q') \end{array}$$

$$\begin{array}{rcl} S & := & \lambda n''.xk'\langle i', n'' \mid w'_l, \lceil a' \rceil, w'_r \mid \lceil q' \rceil \rangle \\ L & = & \lambda n''.w'_l L_0^{q'\ a'} L_1^{q'\ a'} L_\square^{q'\ a'} L_\varepsilon^{q'\ a'} xk'i'n''w'_r : , , , , : , , , , \\ R & = & \lambda n''.w'_r R_0^{q'\ a'} R_1^{q'\ a'} R_\square^{q'\ a'} R^{q'} \\ L^{q'} & & {}_{\varepsilon\ a'}^{\ a'} xk'i'n''w'_l \ {}_{,a''}^{\ a'} := \quad \lambda w'_l.\lambda x.\lambda k'.\lambda i'.\lambda n'.\text{追} \end{array}$$

加$(\lambda w'_r.xk'\langle i', n' \mid w'_l, \lceil a'' \rceil, w'_r \mid \lceil q' \rceil \rangle)_{L^{q'},\varepsilon}{}^{a'} := \quad \lambda x.\lambda k'.\lambda i'.\lambda n'.$追加$^{a'}((\lambda d.\lambda w'_r.xk'\langle i', n' \mid d, \lceil \square \rceil, w'_r \mid \lceil q' \rceil \rangle)\overline{\varepsilon})_{R^{q'},a''}{}^{a'} := \quad \lambda w'_r.\lambda x.\lambda k'.\lambda i'.\lambda n'.$追加$^{a'}(\lambda w'_l.xk'\langle i', n' \mid w'_l, \lceil a'' \rceil, w'_r \mid \lceil q' \rceil \rangle)_{R^{q'},\varepsilon}{}^{a'} := \quad \lambda x.\lambda k'.\lambda i'.\lambda n'.$追加$^{a'}((\lambda d.\lambda w'_l.xk'\langle i', n' \mid w'_l, \lceil \square \rceil, d \mid \lceil q' \rceil \rangle)\overline{\varepsilon})$

3 编码的时间正确性

将输入字符串转换为初始配置。以下引理给出了构造初始配置的项 init。

引理 3.1（将输入字符串转换为初始配置）。设 $\mathcal{M} = (Q, q_{in}, q_T, q_F, \delta)$ 是一台图灵机。存在一个项 init$^{\mathcal{M}}$，或简写为 init，使得对于任意延续项 k 和任意输入字符串 $i \in \mathbb{B}_{\mathsf{I}}^*$（其中$i = \mathrm{L}\cdot s\cdot\mathrm{R}$ 且 $s \in \mathbb{B}^*$）：

$$\texttt{init}\, k\, \overline{i} \quad \rightarrow_{det}^{\Theta(1)} \quad k\, \overline{C_{\texttt{in}}(i)}$$

其中 $C_{\texttt{in}}(i)$ 是 $\mathcal{M}$ 在 i 上的初始配置。

证明。定义

$$\texttt{init} := (\lambda d.\lambda e.\lambda f.\lambda k'.\lambda i'.k'\langle i', d \mid e, \lceil \square \rceil^{\mathbb{B}_{\mathsf{W}}}, f \mid \lceil q_{in} \rceil^{Q} \rangle)\overline{\lfloor 0 \rfloor}\overline{\varepsilon}^{\mathbb{B}_{\mathsf{W}}^*}\overline{\varepsilon}^{\mathbb{B}_{\mathsf{W}}^*}$$

请注意，该项并不处于范式。这是出于技术原因，接下来会对此作出说明。然后

$$\begin{array}{rcl}
\texttt{init}\,k\,\overline{i}^{\mathbb{B}_{\mathsf{I}}^*} & = & (\lambda d.\lambda e.\lambda f.\lambda k'.\lambda i'.k'\langle i', d \mid e, \lceil\Box\rceil^{\mathbb{B}_{\mathsf{W}}}, f \mid \lceil q_{in}\rceil^{Q}\rangle)\overline{\lfloor 0\rfloor}\overline{\varepsilon}^{\mathbb{B}_{\mathsf{W}}^*}\overline{\varepsilon}^{\mathbb{B}_{\mathsf{W}}^*}k\overline{i}^{\mathbb{B}_{\mathsf{I}}^*} \\
& \to_{det}^{5} & k\,\langle \overline{i}^{\mathbb{B}_{\mathsf{I}}^*}, \overline{\lfloor 0\rfloor} \mid \overline{\varepsilon}^{\mathbb{B}_{\mathsf{W}}^*}, \lceil\Box\rceil^{\mathbb{B}_{\mathsf{W}}}, \overline{\varepsilon}^{\mathbb{B}_{\mathsf{W}}^*} \mid \lceil q_{in}\rceil^{Q}\rangle \\
& = & k\,\overline{(i, 0 \mid \varepsilon, \Box, \varepsilon \mid q_{in})} \\
& = & k\,\overline{C_{\texttt{in}}(i)}
\end{array}$$

□

从最终配置中提取输出。

引理 3.2（从最终配置中提取输出）。设 $\mathcal{M} = (Q, q_{in}, q_T, q_F, \delta)$ 是一台图灵机。存在一个项 $\mathsf{final}^{\mathcal{M}}$，或简称 final，使得对于每个延续项 k以及每个状态为 $q \in Q_{fin}$ 的最终配置C：

$$\texttt{final}\,k\,\overline{C} \to_{det}^{\Theta(|Q|)} \begin{cases} k(\lambda x.\lambda y.x) & \text{if } q = q_T \\ k(\lambda x.\lambda y.y) & \text{if } q = q_F \end{cases}$$

证明。定义

$$\texttt{final} := \lambda k'.\lambda C'.C'(\lambda i'.\lambda n'.\lambda w_l'.\lambda a'.\lambda w_r'.\lambda q'.q' N_1 \dots N_{|Q|} k')$$

其中：

$$N_i := \begin{cases} \lambda k'.k'(\lambda x.\lambda y.x) & \text{if } q_i = q_T \\ \lambda k'.k'(\lambda x.\lambda y.y) & \text{if } q_i = q_F \\ \text{whatever closed term (say, the identity)} & \text{otherwise} \end{cases}$$

那么：

$$\begin{array}{rcl}
 & \texttt{final}\,k\,\overline{C} & \\
= & (\lambda k'.\lambda C'.C'(\lambda i'.\lambda n'.\lambda w_l'.\lambda a'.\lambda w_r'.\lambda q'.q' N_1 \dots N_{|Q|} k'))k\overline{C} & \\
\to_{det}^{2} & \overline{C}(\lambda i'.\lambda n'.\lambda w_l'.\lambda a'.\lambda w_r'.\lambda q'.q' N_1 \dots N_{|Q|} k) & \\
= & \overline{(i, n \mid w_l, a, w_r \mid q)}(\lambda i'.\lambda n'.\lambda w_l'.\lambda a'.\lambda w_r'.\lambda q'.q' N_1 \dots N_{|Q|} k) & \\
= & (\lambda x.x\overline{i}^{\mathbb{B}^*}\overline{\lfloor n\rfloor}\ \overline{w_l^{\mathrm{R}}}^{\mathbb{B}_{\mathsf{W}}^*}\ \lceil a\rceil^{\mathbb{B}_{\mathsf{W}}}\ \overline{w_r}^{\mathbb{B}_{\mathsf{W}}^*}\ \lceil q\rceil^{Q})(\lambda i'.\lambda n'.\lambda w_l'.\lambda a'.\lambda w_r'.\lambda q'.q' N_1 \dots N_{|Q|} k) & \\
\to_{det} & (\lambda i'.\lambda n'.\lambda w_l'.\lambda a'.\lambda w_r'.\lambda q'.q' N_1 \dots N_{|Q|} k)\overline{i}^{\mathbb{B}^*}\overline{\lfloor n\rfloor}\ \overline{w_l^{\mathrm{R}}}^{\mathbb{B}_{\mathsf{W}}^*}\ \lceil a\rceil^{\mathbb{B}_{\mathsf{W}}}\ \overline{w_r}^{\mathbb{B}_{\mathsf{W}}^*}\ \lceil q\rceil^{Q} & \\
\to_{det}^{6} & \lceil q\rceil^{Q} N_1 \dots N_{|Q|} k & \\
= & (\lambda x_1 \dots x_{|Q|}.x_j) N_1 \dots N_{|Q|} k & \\
\to_{det}^{|Q|} & N_j k &
\end{array}$$

若 $q = q_T$，则：

$$\begin{array}{rcl}
N_j k & = & (\lambda k'.k'(\lambda x.\lambda y.x))k \\
 & \to_{det} & k(\lambda x.\lambda y.x)
\end{array}$$

若 $q = q_F$，则：

$$\begin{array}{rcl}
N_j k & = & (\lambda k'.k'(\lambda x.\lambda y.y))k \\
 & \to_{det} & k(\lambda x.\lambda y.y)
\end{array}$$

□

模拟机器的一次转移。现在我们证明，将图灵机的转移函数 δ 编码为一个 λ-项后，可以在 $\mathcal{O}(|i|\log |i|)$ 时间内模拟每一次转移，其中 i 是输入字符串。这是编码的核心，也是证明过程中最复杂的部分。

引理 3.3（机器转移的模拟）。设 $\mathcal{M} = (Q, q_{in}, q_T, q_F, \delta)$ 是一台图灵机。项 $\mathsf{trans}^{\mathcal{M}}$ 满足：对于任意延续项 k 以及输入字符串 $i \in \mathbb{B}^+$的任意配置 C：

- 最终配置：若 C是最终配置，则 $\mathsf{trans}\ k\,\overline{C} \to^{\mathcal{O}_{(|i|\log |i|)\, det}} k\,\overline{C}$；
- 非终止配置：若 $C \to_{\mathcal{M}} D$，则 $\mathsf{trans}\ k\,\overline{C} \to^{\mathcal{O}_{(|i|\log |i|)\, det}} \mathsf{trans}\ k\,\overline{D}$。

证明。设 $C = (i, n \mid w_l, a, w_r \mid q)$。下面我们将详细说明 λ 演算如何模拟转移函数。从步数来看，主要开销发生在开始时，来自查找函数：它查找输入字符串 i 中的第 n 个字符。此类调用的开销为$\mathcal{O}(n \log n)$，但由于 n可能变化且 $n \le |i|$，此开销有上界 $\mathcal{O}(|i|\log |i|)$。输入带上的位置不变的转移情形，其开销为常数。输入位置发生变化的情形则需要通过前驱或后继函数来改变计数器 n，这需要 $\mathcal{O}(\log n)$，而其本身又有前述查找开销 $\mathcal{O}(|i|\log |i|)$ 作为上界。

现在，如果 $\textsc{lookup}\,\lfloor n \rfloor i = b$ 那么：

转移 $k\,\overline{C}$=fix 转移aux$k\overline{C}\to^{2}_{det}$转移aux($\lambda z$.fix 转移aux$z$)$k\overline{C}$=转移aux($\lambda z$.转移 z)$k\overline{C}$=($\lambda x.\lambda k'.\lambda C'.C'$($\lambda i'.\lambda n'.\lambda w_l'.\lambda a'.\lambda w_r'.\lambda q'$.查找$\,T\,i'\,n'$))($\lambda z$.转移 z)$k\overline{C}\to^{3}_{det}\overline{C}$ ($\lambda i'.\lambda n'.\lambda w_l'.\lambda a'.\lambda w_r'.\lambda q'$.查找 $T\{x\leftarrow\lambda z.$转移 $z\}\,i'\,n'$)=$(i, n \mid w_l, a, w_r \mid q)$($\lambda i'.\lambda n'.\lambda w_l'.\lambda a'.\lambda w_r'.\lambda q'$.查找 $T\{x\leftarrow\lambda z.$转移 $z\}\,i'\,n'$)=($\lambda x.x\,\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\,\lceil a\rceil\,\overline{w_r}\,\lceil q\rceil$)($\lambda i'.\lambda n'.\lambda w_l'.\lambda a'.\lambda w_r'.\lambda q'$.查找 $T\{x\leftarrow\lambda z.$转移 $z\}\,i'\,n'$)=($\lambda i'.\lambda n'.\lambda w_l'.\lambda a'.\lambda w_r'.\lambda q'$.查找 $T\{x\leftarrow\lambda z.$转移 $z\}\,i'\,n'$)$\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\,\lceil a\rceil\,\overline{w_r}\,\lceil q\rceil\to^{6}_{det}$查找 ($\lambda b'.b'\;A_0A_1A_{\mathsf{L}}A_{\mathsf{R}}\lceil a\rceil\,\lceil q\rceil$($\lambda z.$转移 z)$k'\,\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}\overline{w_r}}$)$\overline{i\lfloor n\rfloor}_{L.\ 2.5}\to^{\mathcal{O}(n\log n)}_{det}$($\lambda b'.b'\;A_0A_1A_{\mathsf{L}}A_{\mathsf{R}}\lceil a\rceil\,\lceil q\rceil$($\lambda z.$转移 z) $k'\,\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}\overline{w_r}}$)$\lceil b\rceil\to_{det}\lceil b\rceil\;A_0A_1A_{\mathsf{L}}A_{\mathsf{R}}\lceil a\rceil\,\lceil q\rceil$($\lambda z.$转移 z)$k\,\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\,\overline{w_r}\to^{4}_{det}A_b\lceil a\rceil\,\lceil q\rceil$($\lambda z.$ 转移 z)$k\,\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\,\overline{w_r}$=($\lambda a'.a'\,B_{b,0}B_{b,1}B_{b,\square}$)$\lceil a\rceil\,\lceil q\rceil$($\lambda z.$转移 z)$k\,\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\,\overline{w_r}\to_{det}\lceil a\rceil B_b$, $_0B_{b,1}B_{b,\square}\lceil q\rceil$($\lambda z.$转移 z)$k\,\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}\overline{w_r}}\to^{3}_{det}B_{b,a}\lceil q\rceil$($\lambda z.$转移 z)$k\,\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}\overline{w_r}}$=($\lambda q'.q'\,C_{b,a,}$ $_{q_1}\ldots C_{b,a,q_{|Q|}}$)$\lceil q\rceil$($\lambda z.$转移 z)$k\,\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}\overline{w_r}}\to_{det}$ $\lceil q\rceil C_{b\;a\;q_1}\ldots C_{b\;a\;q_{|Q|}}\;\lambda z.\qquad z\;k\,\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}\overline{w_r}}$, , , , (转移 $)C_{b,a,q}$($\lambda z.$转移 z) $\to^{|Q|}_{det}$ $k\,\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}\overline{w_r}}$现在，考虑以下四种情况，它们取决于 $\delta(b, a, q)$ 的值：

1. 最终状态：如果 $\delta(b, a, q)$ 未定义，那么 $q \in Q_{fin}$ 并将 $C_{b,\ a,\ q}$替换为对应的 λ-项，我们得到：

$$
\begin{aligned}
&C_{b,a,q}(\lambda z.\mathtt{trans}\, z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}} w_r}\\
&=\\
&(\lambda x.\lambda k'.\lambda i'.\lambda n'.\lambda w_l'.\lambda w_r'.k'\langle i', n' \mid w_l', \lceil a\rceil, w_r' \mid \lceil q\rceil\rangle)(\lambda z.\mathtt{trans}\, z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}} w_r}\\
&\to_{det}^{6}\\
&k\langle \overline{i}, \overline{\lfloor n\rfloor} \mid \overline{w_l^{\mathrm{R}}}, \lceil a\rceil, \overline{w_r} \mid \lceil q\rceil\rangle\\
&=\\
&k\overline{(i, n \mid w_l, a, w_r \mid q)}\\
&=\\
&k\overline{C}
\end{aligned}
$$

2. 读写头不移动：如果 $\delta(b,a,q) = (0 \mid a', \downarrow \mid q')$，那么 $D = (i, n \mid w_l, a', w_r, q')$。模拟继续如下：

$$
\begin{aligned}
&C_{b,a,q}(\lambda z.\mathtt{trans}\, z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}} w_r}\\
&=\\
&(\lambda x.\lambda k'.\lambda i'.\lambda n'.\lambda w_l'.\lambda w_r'.Sn')(\lambda z.\mathtt{trans}\, z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}} w_r}\\
&=\\
&(\lambda x.\lambda k'.\lambda i'.\lambda n'.\lambda w_l'.\lambda w_r'.(\lambda n''.xk'\langle i', n'' \mid w_l', \lceil a'\rceil, w_r' \mid \lceil q'\rceil\rangle)n')(\lambda z.\mathtt{trans}\, z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}} w_r}\\
&\to_{det}^{6}\\
&(\lambda n''.(\lambda z.\mathtt{trans}\, z)k\langle \overline{i}, n'' \mid \overline{w_l^{\mathrm{R}}}, \lceil a'\rceil, \overline{w_r} \mid \lceil q'\rceil\rangle)\overline{\lfloor n\rfloor}\\
&\to_{det}^{2}\\
&\mathtt{trans}\, k\langle \overline{i}, \overline{\lfloor n\rfloor} \mid \overline{w_l^{\mathrm{R}}}, \lceil a'\rceil, \overline{w_r} \mid \lceil q'\rceil\rangle\\
&=\\
&\mathtt{trans}\, k\overline{(i, n \mid w_l, a', w_r \mid q')}\\
&=\\
&\mathtt{trans}\, k\overline{D}
\end{aligned}
$$

3. 输入头不移动，工作头向左移动：如果 $\delta(b,a,q) = (0 \mid a', \leftarrow \mid q')$ 且 $w_l = wa''$，那么：

$$
\begin{aligned}
&C_{b,a,q}(\lambda z.\mathtt{trans}\, z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}} w_r}\\
&=\\
&(\lambda x.\lambda k'.\lambda i'.\lambda n''.\lambda w_l'.\lambda w_r'.Ln')(\lambda z.\mathtt{trans}\, z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}} w_r}\\
&=\\
&(\lambda x.\lambda k'.\lambda i'.\lambda n'.\lambda w_l'.\lambda w_r'.(\lambda n''.w_l' L_0^{q',a'} L_1^{q',a'} L_{\square}^{q',a'} L_{\varepsilon}^{q',a'} xk'i'n''w_r')n')(\lambda z.\mathtt{trans}\, z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}} w_r}\\
&\to_{det}^{6}\\
&(\lambda n''.\overline{w_l^{\mathrm{R}}} L_0^{q',a'} L_1^{q',a'} L_{\square}^{q',a'} L_{\varepsilon}^{q',a'} (\lambda z.\mathtt{trans}\, z)k\overline{in''w_r})\overline{\lfloor n\rfloor}\\
&\to_{det}\\
&\overline{w_l^{\mathrm{R}}} L_0^{q',a'} L_1^{q',a'} L_{\square}^{q',a'} L_{\varepsilon}^{q',a'} (\lambda z.\mathtt{trans}\, z)k\overline{i\lfloor n\rfloor w_r}
\end{aligned}
$$

分为两种子情况，取决于 w_l 是空字符串还是复合字符串。

(a) w_l 是复合字符串 wa''。接着，$w_l^{\mathrm{R}} = a''w^{\mathrm{R}}$ 和 $D = (i,\ n \mid w_l,\ a'',\ a'w_r \mid q')$。模拟继续如下：

$$
\begin{aligned}
&= \\
&\overline{a''w^{\mathrm{R}}} L_0^{q',a'} L_1^{q',a'} L_{\square}^{q',a'} L_{\varepsilon}^{q',a'} (\lambda z.\texttt{trans}\, z) k \overline{i \lfloor n \rfloor} \overline{w_r} \\
&\to_{det}^{4} \\
&L_{a''}^{q',a'} \overline{w^{\mathrm{R}}} (\lambda z.\texttt{trans}\, z) k \overline{i \lfloor n \rfloor} \overline{w_r} \\
&= \\
&(\lambda w_l'.\lambda x.\lambda k'.\lambda i'.\lambda n'.\texttt{append}^{a'} (\lambda w_r'.x k' \langle i', n' \mid w_l', \lceil a'' \rceil, w_r' \mid \lceil q' \rceil \rangle)) \overline{w^{\mathrm{R}}} (\lambda z.\texttt{trans}\, z) k \overline{i \lfloor n \rfloor} \overline{w_r} \\
&\to_{det}^{5} \\
&\texttt{append}^{a'} (\lambda w_r'.(\lambda z.\texttt{trans}\, z) k \langle \overline{i}, \overline{\lfloor n \rfloor} \mid \overline{w^{\mathrm{R}}}, \lceil a'' \rceil, w_r' \mid \lceil q' \rceil \rangle) \overline{w_r} \\
&L.\ 2.2 \to_{det}^{O(1)} \\
&(\lambda w_r'.(\lambda z.\texttt{trans}\, z) k \langle \overline{i}, \overline{\lfloor n \rfloor} \mid \overline{w^{\mathrm{R}}}, \lceil a'' \rceil, w_r' \mid \lceil q' \rceil \rangle) \overline{a' w_r} \\
&\to_{det}^{2} \\
&\texttt{trans}\, k \langle \overline{i}, \overline{\lfloor n \rfloor} \mid \overline{w^{\mathrm{R}}}, \lceil a'' \rceil, \overline{a' w_r} \mid \lceil q' \rceil \rangle \\
&= \\
&\texttt{trans}\, k \overline{(i, n \mid w, a'', a' w_r \mid q')} \\
&= \\
&\texttt{trans}\, k \overline{D}
\end{aligned}
$$

(b) w_l 是空字符串 ε。接着， $D = (i,\ n \mid \varepsilon,\ \square,\ a' w_r \mid q')$ 。模拟继续如下:

$$
\begin{aligned}
&= \\
&\overline{\varepsilon} L_0^{q',a'} L_1^{q',a'} L_{\square}^{q',a'} L_{\varepsilon}^{q',a'} (\lambda z.\texttt{trans}\, z) k \overline{i \lfloor n \rfloor} \overline{w_r} \\
&\to_{det}^{4} \\
&L_{\varepsilon}^{q',a'} (\lambda z.\texttt{trans}\, z) k \overline{i \lfloor n \rfloor} \overline{w_r} \\
&= \\
&(\lambda x.\lambda k'.\lambda i'.\lambda n'.\texttt{append}^{a'} ((\lambda d.\lambda w_r.x k' \langle i', n' \mid d, \lceil \square \rceil, w_r \mid \lceil q' \rceil \rangle) \overline{\varepsilon})) (\lambda z.\texttt{trans}\, z) k \overline{i \lfloor n \rfloor} \overline{w_r} \\
&\to_{det}^{5} \\
&\texttt{append}^{a'} ((\lambda d.\lambda w_r'.(\lambda z.\texttt{trans}\, z) k \langle \overline{i}, \overline{\lfloor n \rfloor} \mid d, \lceil \square \rceil, w_r' \mid \lceil q' \rceil \rangle) \overline{\varepsilon}) \overline{w_r} \\
&L.\ 2.2 \to_{det}^{O(1)} \\
&((\lambda d.\lambda w_r'.(\lambda z.\texttt{trans}\, z) k \langle \overline{i}, \overline{\lfloor n \rfloor} \mid d, \lceil \square \rceil, w_r' \mid \lceil q' \rceil \rangle) \overline{\varepsilon}) \overline{a' w_r} \\
&\to_{det}^{3} \\
&\texttt{trans}\, k \langle \overline{i}, \overline{\lfloor n \rfloor} \mid \overline{\varepsilon}, \lceil \square \rceil, \overline{a' w_r} \mid \lceil q' \rceil \rangle \\
&= \\
&\texttt{trans}\, k \overline{(i, n \mid \varepsilon, \square, a' w_r \mid q')} \\
&= \\
&\texttt{trans}\, k \overline{D}
\end{aligned}
$$

4. 输入头不移动，工作头向右移动：若 $\delta(b, a, q) = (0 \mid a', \to \mid q')$ 和 $w_r = a'' w$，则:

$C_{b,a,q}(\lambda z.\text{转移}\ z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\overline{w_r}=(\lambda x.\lambda k'.\lambda i'.\lambda n'.\lambda w_l'.\lambda w_r'.Rn')(\lambda z.\text{转移}\ z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\overline{w_r}=(\lambda x.\lambda k'.\lambda i'.\lambda n'.\lambda w_l'.\lambda w_r'.(\lambda n''.w_r'R^{q'},{}_0{}^{a'}R^{q'},{}_1{}^{a'}R^{q'},{}_\square{}^{a'}R^{q'},{}_\varepsilon{}^{a'}xk'i'n''w_l')n')(\lambda z.\text{转移}\ z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\overline{w_r}\to_{det}^{6}(\lambda n''.\overline{w_r}R^{q'},{}_0{}^{a'}R^{q'},{}_1{}^{a'}R^{q'},{}_\square{}^{a'}R^{q'},{}_\varepsilon{}^{a'}(\lambda z.\text{转移}\ z)k\overline{i}n''\overline{w_l^{\mathrm{R}}})\overline{\lfloor n\rfloor}\to_{det}\overline{w_r}R^{q'},{}_0{}^{a'}R^{q'},{}_1{}^{a'}R^{q'},{}_\square{}^{a'}R^{q'},{}_\varepsilon{}^{a'}(\lambda z.\text{转移}\ z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}$

根据 w_l 是空字符串还是复合字符串，分为两种子情况。

(a) w_r 是复合字符串 $a''w$。那么 $D=(i,n\mid w_la',a'',w\mid q')$。模拟继续如下：

$$
\begin{aligned}
&=\\
&\overline{a''w}R_0^{q',a'}R_1^{q',a'}R_\square^{q',a'}R_\varepsilon^{q',a'}(\lambda z.\mathtt{trans}\ z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\\
&\to_{det}^{4}\\
&R_{a''}^{q',a'}\overline{w}(\lambda z.\mathtt{trans}\ z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\\
&=\\
&(\lambda w_r'.\lambda x.\lambda k'.\lambda i'.\lambda n'.\mathtt{append}^{a'}(\lambda w_l'.xk'\langle i',n'\mid w_l',\lceil a''\rceil,w_r'\mid\lceil q'\rceil\rangle))\overline{w}(\lambda z.\mathtt{trans}\ z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\\
&\to_{det}^{5}\\
&\mathtt{append}^{a'}(\lambda w_l'.(\lambda z.\mathtt{trans}\ z)k\langle\overline{i},\overline{\lfloor n\rfloor}\mid w_l',\lceil a''\rceil,w\mid\lceil q'\rceil\rangle)\overline{w_l^{\mathrm{R}}}\\
&L.\ 2.2\to_{det}^{O(1)}(\lambda w_l'.(\lambda z.\mathtt{trans}\ z)k\langle\overline{i},\overline{\lfloor n\rfloor}\mid w_l',\lceil a''\rceil,w\mid\lceil q'\rceil\rangle)\overline{a'w_l^{\mathrm{R}}}\\
&\to_{det}^{2}\\
&\mathtt{trans}\ k\langle\overline{i},\overline{\lfloor n\rfloor}\mid\overline{a'w_l^{\mathrm{R}}},\lceil a''\rceil,w\mid\lceil q'\rceil\rangle\\
&=\\
&\mathtt{trans}\ k\langle\overline{i},\overline{\lfloor n\rfloor}\mid\overline{(w_la')^{\mathrm{R}}},\lceil a''\rceil,w\mid\lceil q'\rceil\rangle\\
&=\\
&\mathtt{trans}\ k\overline{(i,n\mid w_la',a'',w\mid q')}\\
&=\\
&\mathtt{trans}\ k\overline{D}
\end{aligned}
$$

(b) w_r 是空字符串 ε。于是 $D=(i,n\mid w_la',\square,\varepsilon\mid q')$。模拟继续如下：

$$
\begin{aligned}
&= \\
&\overline{\varepsilon} R_0^{q',a'} R_1^{q',a'} R_\square^{q',a'} R_\varepsilon^{q',a'} (\lambda z.\texttt{trans}\, z) k \overline{i} \overline{\lfloor n \rfloor} \overline{w_l^{\mathrm{R}}} \\
&\to_{det}^{4} \\
&R_\varepsilon^{q',a'} (\lambda z.\texttt{trans}\, z) k \overline{i} \overline{\lfloor n \rfloor} \overline{w_l^{\mathrm{R}}} \\
&= \\
&(\lambda x.\lambda k'.\lambda i'.\lambda n'.\texttt{append}^{a'} ((\lambda d.\lambda w_l'.x k' \langle i', n' \mid w_l', \lceil \square \rceil, d \mid \lceil q' \rceil \rangle) \overline{\varepsilon})) (\lambda z.\texttt{trans}\, z) k \overline{i} \overline{\lfloor n \rfloor} \overline{w_l^{\mathrm{R}}} \\
&\to_{det}^{5} \\
&\texttt{append}^{a'} ((\lambda d.\lambda w_l'.(\lambda z.\texttt{trans}\, z) k \langle \overline{i}, \overline{\lfloor n \rfloor} \mid w_l', \lceil \square \rceil, d \mid \lceil q' \rceil \rangle) \overline{\varepsilon}) \overline{w_l^{\mathrm{R}}} \\
&L.\ 2.2 \to_{det}^{O(1)} \\
&((\lambda d.\lambda w_l'.(\lambda z.\texttt{trans}\, z) k \langle \overline{i}, \overline{\lfloor n \rfloor} \mid w_l', \lceil \square \rceil, d \mid \lceil q' \rceil \rangle) \overline{\varepsilon}) \overline{a' w_l^{\mathrm{R}}} \\
&\to_{det}^{3} \\
&\texttt{trans}\, k \langle \overline{i}, \overline{\lfloor n \rfloor} \mid \overline{a' w_l^{\mathrm{R}}}, \lceil \square \rceil, \overline{\varepsilon} \mid \lceil q' \rceil \rangle) \\
&= \\
&\texttt{trans}\, k \langle \overline{i}, \overline{\lfloor n \rfloor} \mid \overline{(w_l a')^{\mathrm{R}}}, \lceil \square \rceil, \overline{\varepsilon} \mid \lceil q' \rceil \rangle) \\
&= \\
&\texttt{trans}\, k \overline{(i, n \mid w_l a', \square, \varepsilon \mid q')} \\
&= \\
&\texttt{trans}\, k \overline{D}
\end{aligned}
$$

5. 输入头向左移动，工作头不动：如果 $\delta(b, a, q) = (-1 \mid a', \downarrow \mid q')$，那么 $D = (i, n-1 \mid w_l, a', w_r, q')$。模拟继续如下：

$$
\begin{aligned}
&C_{b,a,q} (\lambda z.\texttt{trans}\, z) k \overline{i} \overline{\lfloor n \rfloor} \overline{w_l^{\mathrm{R}}} \overline{w_r} \\
&= \\
&(\lambda x.\lambda k'.\lambda i'.\lambda n'.\lambda w_l'.\lambda w_r'.\texttt{pred}\, S (\lambda z.\texttt{trans}\, z) k \overline{i} \overline{\lfloor n \rfloor} \overline{w_l^{\mathrm{R}}} \overline{w_r} \\
&= \\
&(\lambda x.\lambda k'.\lambda i'.\lambda n'.\lambda w_l'.\lambda w_r'.\texttt{pred}\, (\lambda n''.x k' \langle i', n'' \mid w_l', \lceil a' \rceil, w_r' \mid \lceil q' \rceil \rangle) n') (\lambda z.\texttt{trans}\, z) k \overline{i} \overline{\lfloor n \rfloor} \overline{w_l^{\mathrm{R}}} \overline{w_r} \\
&\to_{det}^{6} \\
&\texttt{pred}\, (\lambda n''.(\lambda z.\texttt{trans}\, z) k \langle \overline{i}, n'' \mid \overline{w_l^{\mathrm{R}}}, \lceil a' \rceil, \overline{w_r} \mid \lceil q' \rceil \rangle) \overline{\lfloor n \rfloor} \\
&L.\ 2.4 \to_{det}^{\mathcal{O}(\log n)} \\
&(\lambda n''.(\lambda z.\texttt{trans}\, z) k \langle \overline{i}, n'' \mid \overline{w_l^{\mathrm{R}}}, \lceil a' \rceil, \overline{w_r} \mid \lceil q' \rceil \rangle) \overline{\lfloor n-1 \rfloor} \\
&= \\
&\texttt{trans}\, k \overline{(i, \lfloor n-1 \rfloor \mid w_l, a', w_r \mid q')} \\
&= \\
&\texttt{trans}\, k \overline{(i, \lfloor n-1 \rfloor \mid w_l, a', w_r \mid q')} \\
&= \\
&\texttt{trans}\, k \overline{D}
\end{aligned}
$$

6. 输入头向左移动，工作头也向左移动：如果 $\delta(b, a, q) = (-1 \mid a', \leftarrow \mid q')$ 且 $w_l = w a''$，那么 $D = (i, n-1 \mid w, a'', a' w_r, q')$。模拟继续如下：

$C_{b,a,q}(\lambda z.\text{转移}\ z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\overline{w_r}=(\lambda x.\lambda k'.\lambda i'.\lambda n'.\lambda w_l'.\lambda w_r'.\text{前驱}\ Ln')(\lambda z.\text{转移}\ z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\overline{w_r}=(\lambda x.\lambda k'.\lambda i'.\lambda n'.\lambda w_l'.\lambda w_r'.\text{前驱}\ (\lambda n''.w_l'L^{q'}_{\ a'}L^{q'}_{\ a'}L^{q'}_{\square\ a'}L^{q'}_{\varepsilon\ a'}xk'i'n''w_r')n')(\lambda z.\text{转移}\ z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\overline{w_r}\to^{6}_{det}\text{前驱}\ (\lambda n''.\overline{w_l^{\mathrm{R}}}L^{q'}_{0\ a'}L^{q'}_{1\ a'}L^{q'}_{\square\ a'}L^{q'}_{\varepsilon\ a'}\lambda z.\ z\ k\bar{i}n''\overline{w_r}\ \overline{\lfloor n\rfloor},,,,(\text{转移}))$

$L.\ 2.4\to^{\mathcal{O}(\log n)}_{det}$

$(\lambda n''.\overline{w_l^{\mathrm{R}}}L^{q'}_{\ a'}L^{q'}_{0\ a'}L^{q'}_{1\ a'}L^{q'}_{\square\ a'}L^{q'}_{\varepsilon\ a'}(\lambda z.\text{转移}\ z)k\bar{i}n''\overline{w_r})\overline{\lfloor n-1\rfloor}\to_{det}\overline{w_l^{\mathrm{R}}}L^{q'}_{0\ a'}L^{q'}_{1\ a'}L^{q'}_{\square\ a'}L^{q'}_{\varepsilon\ a'}(\lambda z.\text{转移}\ z)k\bar{i}\overline{\lfloor n-1\rfloor w_r}$

接着，情形继续分为情形 3 的两个子情况（输入头不移动，工作头向左移动），唯一的区别是将$\overline{\lfloor n\rfloor}$替换为$\overline{\lfloor n-1\rfloor}$。

7. 输入头向左移动，工作头向右移动：如果 $\delta(b,a,q)=(-1\mid a',\leftarrow\mid q')$ 且 $w_l=wa''$，那么 $D=(i,n-1\mid w,a'',a'w_r,q')$。模拟如下继续进行：

$C_{b,a,q}(\lambda z.\texttt{trans}\ z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\overline{w_r}$
$=$
$(\lambda x.\lambda k'.\lambda i'.\lambda n'.\lambda w_l'.\lambda w_r'.\texttt{pred}\,Rn')(\lambda z.\texttt{trans}\ z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\overline{w_r}$
$=$
$(\lambda x.\lambda k'.\lambda i'.\lambda n'.\lambda w_l'.\lambda w_r'.\texttt{pred}(\lambda n''.w_r'R_0^{q',a'}R_1^{q',a'}R_\square^{q',a'}R_\varepsilon^{q',a'}xk'i'n''w_l')n')(\lambda z.\texttt{trans}\ z)k\overline{i\lfloor n\rfloor w_l^{\mathrm{R}}}\overline{w_r}$
$\to^{6}_{det}$
$\texttt{pred}(\lambda n''.\overline{w_r}R_0^{q',a'}R_1^{q',a'}R_\square^{q',a'}R_\varepsilon^{q',a'}(\lambda z.\texttt{trans}\ z)k\bar{i}n''\overline{w_l^{\mathrm{R}}})\overline{\lfloor n\rfloor}$
by $L.\ 2.4$ $\to^{\mathcal{O}(\log n)}_{det}$
$(\lambda n''.\overline{w_r}R_0^{q',a'}R_1^{q',a'}R_\square^{q',a'}R_\varepsilon^{q',a'}(\lambda z.\texttt{trans}\ z)k\bar{i}n''\overline{w_l^{\mathrm{R}}})\overline{\lfloor n-1\rfloor}$
$\to_{det}$
$\lambda n''.\overline{w_r}R_0^{q',a'}R_1^{q',a'}R_\square^{q',a'}R_\varepsilon^{q',a'}(\lambda z.\texttt{trans}\ z)k\overline{i\lfloor n-1\rfloor w_l^{\mathrm{R}}}$

接着，情形继续分为情形 4 的两个子情况（输入头不移动，工作头向右移动），唯一的区别是将$\overline{\lfloor n\rfloor}$替换为$\overline{\lfloor n-1\rfloor}$。

8. 输入头向右移动，工作头不移动：与情形 5（输入头向左，工作头不移动）完全相同，只需将前驱替换为后继函数，并使用引理 2.3 而非引理 2.4。

9. 输入头向右移动，工作头向左移动：与情形 6（输入头向左、工作头向左）完全相同，只需将前驱替换为后继函数，并使用引理 2.3 而非引理 2.4。

10. 输入头向右移动，工作头向右移动：与情形 7（输入头向右、工作头向右）完全相同，只需将前驱替换为后继函数，并使用引理 2.3 而非引理 2.4。 □

对执行长度进行直接归纳，可得以下推论。

推论 3.4（执行）。设 $\mathcal{M}$ 为一台图灵机。则存在一个项 trans，将 $\mathcal{M}$ 编码为引理 3.3 所述形式，使得对于输入字符串 $i \in \mathbb{B}^+$的每个配置C

1. 有限计算：如果 D 是可从 C 经 n 个转移步到达的最终配置，那么存在一个推导 ρ，使得 ρ：trans $k\lceil C\rceil \to_{det}^{\mathcal{O}((n+1)|i|\log|i|)} k\lceil D\rceil$；2. 发散计算：如果从 C 无法到达任何最终配置，那么 trans $k\lceil C\rceil$ 发散。

模拟定理。现在，我们已经具备了证明本文最后一个定理所需的全部要素。

定理 3.5（模拟）。设 $f : \mathbb{B}^* \to \mathbb{B}$ 是由图灵机 $\mathcal{M}$ 在时间 $T_{\mathcal{M}}$ 内计算的函数。那么存在一个将 $\mathbb{B}$、字符串和 $\mathbb{B}$ 上的图灵机编码到 $\Lambda_{\mathtt{det}}$ 中的编码 $\lceil\cdot\rceil$，使得对于每个 $i \in \mathbb{B}^+$，都存在 ρ，满足 $\rho : \lceil\mathcal{M}\rceil\lceil i\rceil \to_{det}^{n} \lceil f(i)\rceil$，其中 $n = \Theta((T_{\mathcal{M}}(|i|) + 1) \cdot |i| \cdot \log |i|)$。

证明。直观地说，该项就是

$$\overline{\mathcal{M}} := \mathtt{init}(\mathtt{trans}(\mathtt{final}\mathsf{I}))$$

其中，恒等项 I 扮演初始延续的角色。

然而，这样的项不属于确定性 λ-演算，因为应用的右子项并不总是值。解决办法很简单，只需对参数进行 η-扩展。因此，定义

$$\overline{\mathcal{M}} := \mathtt{init}(\lambda y.\mathtt{trans}(\lambda x.\mathtt{final}\mathsf{I}x)y)$$

Then

$\overline{\mathcal{M}}\lceil i\rceil$ =初始化$(\lambda y.$转移$(\lambda x.$finalI$x)y)$
$\lceil i\rceil \to_{det}^{\Theta(1)}$ (由 $L.$ 3.1) $(\lambda y.$转移$(\lambda x.$finalI$x)y)\lceil C_{\mathtt{in}}^{\mathcal{M}}(s)\rceil \to_{det}$ 转移$(\lambda x.$ finalI$x)\lceil C_{\mathtt{in}}^{\mathcal{M}}(s)\rceil \to_{det}^{\Theta((T_{\mathcal{M}}(|i|)+1)\cdot|i|\cdot$日志$|i|)}$ (由 $Cor.$ 3.4) $(\lambda x.$ finalI$x)\lceil C_{\mathtt{fin}}(f(i))\rceil \to_{det}$finalI$\lceil C_{\mathtt{fin}}(f(i))$
$\rceil \to_{det}^{\Theta(|Q|)}$ (由 $L.$ 3.2) I$\lceil f(i)$
$\rceil \to_{det} \lceil f(i)\rceil$

□

参考文献

[DLA17] Ugo Dal Lago 和 Beniamino Accattoli。《将图灵机编码进确定性 λ 演算》。CoRR，abs/1711.10078，2017 年。

[Gol00] Mayer Goldberg。《理论珍珠：λ 演算中一种充分且高效的左结合二进制数字系统》。J. Funct. Program., 10(6):607–623，2000年。

[Mog01] Torben Æ. Mogensen。《纯 λ 演算中紧凑高效数字表示的研究》。载于 Dines Bjørner、Manfred Broy 和 Alexandre V. Zamulin 编，《系统信息学展望：第 4 届 Andrei Ershov 纪念国际会议（PSI 2001）》，俄罗斯新西伯利亚学术城，2001年7月2日至6日，修订论文集，计算机科学讲义第 2244 卷，第 205–213 页。Springer，2001年。

[vEB12] Peter van Emde Boas. 面向初学者的图灵机——为何表示方式确实重要。载于 M´aria Bielikov´a、Gerhard Friedrich、Georg Gott- lob、Stefan Katzenbeisser 和 Gy¨orgy Tur´an 编，SOFSEM 2012：计算机科学理论与实践——第38届计算机科学理论与实践当前趋势会议，捷克共和国 Špindler ˚uv Ml´yn，2012年1月21日至27日。会议论文集，计算机科学讲义第7147卷，第14–30页。Springer，2012年。