从双推出图文法到带有和不带指数模态的超图兰贝克文法

Tikhon Pshenitsyn

莫斯科国立罗蒙诺索夫大学数学与力学系数理逻辑与算法理论系，GSP-1，Leninskie Gory，莫斯科，119991，俄罗斯联邦；ptihon at yandex.ru

我们研究如何关联基于双推出（DPO）方法的著名超图文法与基于超图兰贝克演算 HL 的文法（称为 HL 文法）。事实证明，可以采用与 Kanazawa 用于乘法指数线性逻辑的方法类似的手段，将 DPO 规则自然地编码为 HL 的类型。为推广其推理，我们通过添加指数模态扩展超图兰贝克演算，得到一种新的演算 HMEL_0；随后证明，任何 DPO 文法都可以转换为与之等价的 HMEL_0 文法。我们还定义了合取 Kleene 星，其行为与该指数模态类似，并建立了类似的结果。如果既不向 HL 添加指数模态，也不添加合取 Kleene 星，我们仍可使用相同的编码，并证明任何推导长度受线性限制的 DPO 文法都可以转换为与之等价的 HL 文法。

1 引言

本文旨在建立两类图文法之间的联系：双推出（DPO）超图文法和超图兰贝克文法（HL 文法）。

DPO 超图文法是最广为人知的图文法之一，由 1973 [2] 引入。其旨在将无限制的乔姆斯基形式文法从字符串推广到图。回顾一下，形式文法中形如 $\alpha \Rightarrow \beta$ 的产生式允许在任意字符串 γ 中用字符串 β 替换子串 α。而 DPO 超图文法的产生式则可以表示为 $L \Rightarrow R$，其中 L 和 R 是两个超图。然而，用一个超图替换另一个超图的过程还需要进一步说明；这可以通过双推出方法实现，该方法在图文法领域得到了广泛应用。

超图兰贝克演算 HL 和超图兰贝克文法是 [13, 15] 中介绍的新颖方法。它们以逻辑为基础：HL 将 [8] 中提出的 Lambek演算加以推广。兰贝克演算 L 是直觉主义逻辑的一种次结构逻辑，最初旨在为自然语言的句法建模。超图兰贝克演算 HL 继承了 L 的主要原则，以及其结构性质和模型论性质。此外，HL 是超图兰贝克文法（HL 文法）的基础。HL 文法通过将 HL 中有限个类型（即公式）赋给字母表中的符号来定义。略作简化后，这类文法的机制可描述如下：要检验超图 H 是否由某个 HL 文法生成，必须将 H 中的每个符号替换为与之对应的一个类型（从而得到一个带类型标记的超图），并检验所得结构是否可由 HL 的规则从公理导出。

*本论文得到俄罗斯基础研究基金会（RFBR）项目 N20-01-00670、莫斯科大学“脑、认知系统与人工智能”跨学科科学与教育学院以及“BASIS”理论物理与数学发展基金会的资助。

C. Grabmayer（编）：TERMGRAPH 2022 会后论文集，
EPTCS 377，2023，第 9–26 页，doi:10.4204/EPTCS.377.1

© T. Pshenitsyn 本作品采用知识共享署名许可协议授权。

我们的目标是弄清 HL 文法能够生成哪一类超图语言。尤其重要的是将其与受到广泛研究的 DPO 文法进行比较。由此产生了一个问题：能否将每个 DPO 文法转换为生成相同语言的 HL 文法？一个简单的复杂性论证表明，答案是否定的：DPO 文法是图灵完备的，而 HL 文法的成员资格问题是可判定的，甚至是 NP 完全的。尽管如此，事实证明，有一种简单的方法可以将任意 DPO 规则自然地编码为 HL 的一种类型。这种编码本质上与 Kanazawa 定义的编码相同，后者用于将无约束 Chomsky 文法转换为乘法-指数 Lambek 演算上的文法 [4]。在超图情形下，我们还可以为 HL 加入指数模态 !，从而得到超图乘法-指数 Lambek 演算；本文以某种受限的方式定义了该演算，这有助于我们证明其切割消去定理，同时又足以满足我们的目的。这个新演算记作 HMEL_0。接着，我们可以证明，任意 DPO 文法都能转换为 HMEL_0 文法。事实上，HMEL_0 文法显然与 DPO 文法等价；然而，本文只证明了这一说法的一半，因此另一半在形式上仍只是一个猜想。

然而，我们的主要目标仍然是研究不带指数模态的超图兰贝克演算本身。事实证明，使用与 HMEL_0-文法相同的编码，我们可以建立一种良好的关系；为此，我们需要对 DPO 文法加以限制。具体而言，设一个 DPO 文法 Gr 配备一个数 $c \in \mathbb{N}$，并考虑由 Gr 生成的超图语言 H，其中 H 在 Gr 中的推导长度小于或等于 $c \cdot |E_H|$，这里 $|E_H|$ 是 H 中超边的数量。也就是说，相对于超图中超边的数量，我们对其推导长度施加线性限制。事实证明，每个这样的语言都可以由 HL 文法生成。事实上，反向命题（从 HL 文法到受线性限制的 DPO 文法）同样成立；不过，这将是另一篇待撰写文章的主题。

在这篇论文的第一个版本中，我们在海法举行的 TERMGRAPH 2022 上介绍了对 HL 的扩展，采用合取 Kleene 星，而非指数模态。合取 Kleene 星的行为类似于指数模态 !（更准确地说，类似于 [5] 中的软次指数）。不过，合取 Kleene 星的动机不同，其公理化尤其涉及 ω 规则（见第 5 节）。这种运算相当特殊且古怪，在文献中鲜少受到关注（不过，在讨论存储算子时，例如 [10] 中曾提及它）。在研讨会讨论期间，有人建议我尝试使用指数模态来实现相同目标，因为这种做法更为常见，且已有充分研究。随后，在我们开始研究关于字符串的乘法-指数 Lambek 演算的现有文献时，发现了 Kanazawa 的文章 [4]，其中采用了与本研究所提出的构造类似的方法。因此，在论文的最终版本中，我们不再使用合取 Kleene 星来构造文法，而改用指数模态。不过，为了避免过多改变论文内容，我们仍在第 5 节保留了前者的定义。

本文组织如下。第 2 节给出了所有相关形式体系和概念的定义（超图、超边替换、DPO 文法、HL、HL 文法），并介绍了超图兰贝克演算背后的一些直观理解。第 3 节定义超图乘法指数兰贝克演算 HMEL_0，并将 [4] 中的思想推广，从而证明 DPO 文法可以转换为等价的 HMEL_0 文法。第 4 节对 DPO 文法施加线性限制，并给出从受限文法到 HL 文法的翻译过程。第 5 节给出合取 Kleene 星的定义，并对其作出一些评论（尽管该运算在后续转换中不再使用）。第 6 节作出总结。

2 预备知识

Σ^* 是字母表 Σ 上包含空字 Λ 的字符串集合；如果 R 是一个关系，那么 R^* 是其自反传递闭包。每个函数 $f: \Sigma \to \Delta$ 都可以扩展为同态 $f: \Sigma^* \to \Delta^*$。我们用 w(i) 表示 $w \in \Sigma^*$ 的第 i 个符号，并用 $|w|$ 表示 w 中符号的数量。

令 $[n]$ 表示集合 $\{1, 2, \ldots, n\}$（相应地，$[0] := /0$）。

给定标签集 Σ 以及秩函数 $rk: \Sigma \to \mathbb{N}$，定义在 Σ 上的超图 G 是一个元组 $G = \langle V_G, E_G, att_G, lab_G, ext_G \rangle$，其中 V_G 是有限节点集，E_G 是有限超边集，$att_G: E_G \to V_G^*$ 为每条超边指定一个附着节点字符串（将其视为有序多重集），$lab_G: E_G \to \Sigma$ 为每条超边指定一个标签集 Σ 中的元素，且满足：对于任意 $e \in E_G$，$rk(lab_G(e)) = |att_G(e)|$；$ext_G \in V_G^*$ 是由外部节点组成的字符串。超图总是视同构而定。所有标签来自 Σ 的超图构成的集合记为 $\mathscr{H}(\Sigma)$。注意，我们允许一条超边的附着节点彼此重合，也允许外部节点彼此重合。秩函数 rk_G（若 G 明确，则记作 rk）定义如下：$rk_G(e) := |att_G(e)|$。此外，$rk(G) := |ext_G|$。

在超图示意图中，黑色圆圈表示节点，带标签的矩形表示超边，att 用带编号的线表示，外部节点则用括号（圆括号）中的数字表示。如果一条超边恰有两个附着节点，则用一条带标签的箭头表示，该箭头从第一个附着节点指向第二个附着节点。

柄 $a^\bullet$ 是一个超图 $a^\bullet = \langle [n], [1], att, lab\ 1 \ldots n \rangle$，其中 $att(1) = 1 \ldots n$ 且 $lab(1) = a$（$a \in \Sigma$，$rk(a) = n$）。超图 a° 的形式为 $\langle [n], [1], att, lab\ \Lambda \rangle$，其中 att、lab 的定义与 $a^\bullet$ 中相同。超图 $D[k] = \langle [k], /0, /0, /0, \Lambda \rangle$ 称为离散的（$k \in \mathbb{N}$）。

给定超图 H 和函数 $f: E_H \to \Sigma$，重新标记后的超图 f (H) 为 $= \langle V_H, E_H, att_H, f, ext_H \rangle$。要求对 $e \in E_H$，有 $rk_H(e) = rk(f(e))$。

用超图 H 替换 G 中的超边 e_0（满足 $rk(e_0) = rk(H)$）的步骤如下：(1) 从 G 中移除 e_0；(2) 插入 H 的一个同构副本（H 和 G 必须由互不相交的节点集和超边集构成）；(3) 对每个 $i = 1, \ldots, rk(e_0)$，将 H 的第 i 个外部节点与 e_0 的第 i 个附着节点融合。结果记作 $G[e_0/H]$。众所周知，如果将一个超图的若干条超边替换为其他超图，那么结果与替换顺序无关；此外，同时进行替换也不会改变结果 [1]。使用以下记法：若超图 H 中的 $e1, \ldots, e_k$ 是互不相同的超边，且分别同时替换为超图 $H1, \ldots, H_k$，则结果记作 $H[e_1/H1, \ldots, e_k/H_k]$。

若超图 G 没有外部节点（$ext_G = \Lambda$），则称其为零秩。如果超图 H_1, H_2 中至少有一个是零秩的，那么可以将它们的不交并 $H_1 + H_2$ 定义为超图 $\langle V_{H_1} \sqcup V_{H_2}, E_{H_1} \sqcup E_{H_2}, att, lab, ext \rangle$，满足 $att|_{H_i} = att_{H_i}$，$lab|_{H_i} = lab_{H_i}$（$i = 1, 2$），并且当 H_i 不是零秩时 $ext = ext_{H_i}$，否则为 $ext = \Lambda$；也就是说，我们只是将这些超图放在一起，不融合任何节点或超边。零秩超图 H 与自身作 k 次不交并，记作 $k \cdot H$。

2.1 DPO 文法

给定两个超图 G 和 H，态射 $f: G \to H$ 是一对函数 $f_V: V_G \to V_H$、$f_E: E_G \to E_H$，满足对所有 $e \in E_G$，$f_V(att_G(e)) = att_H(f_E(e))$，$lab_H(f_E(e)) = lab_G(e)$，且 $f_V(ext_G) = ext_H$。

设 I、G_1, G_2 为零秩超图，且具有态射 $\varphi_i: I \to G_i$, $i = 1, 2$。令 $\equiv_V$ 为不交并 $V_{G_1} \sqcup V_{G_2}$ 上的最小等价关系，满足对于 $v \in V_I$，$\varphi_1(v) \equiv \varphi_2(v)$；在 $E_{G_1} \sqcup E_{G_2}$ 上也类似地定义关系 $\equiv_E$。若 x 是节点，则 $\langle x \rangle$ 表示 x 关于 $\equiv_V$ 的等价类；若 x 是超边，则表示 x 关于 $\equiv_E$ 的等价类。G_1 和 G_2 沿 I 的粘合记作 $G_1 +_{\varphi_1, \varphi_2} G_2$，它是一个

超图 G，满足 $V_G = (V_{G_1} \sqcup V_{G_2})/\equiv_V$、$E_G = (E_{G_1} \sqcup E_{G_2})/\equiv_E$；给定 $\langle e\rangle \in E_G$，且 $rk(e) = k$，若 $e \in E_{G_i}$，其中某个 $i = 1, 2$，则 $att_G(\langle e\rangle) = \langle att_{G_i}(e)(1)\rangle \ldots \langle att_{G_i}(e)(k)\rangle$，且 $lab_G(\langle e\rangle) = lab_{G_i}(e)$。这是一个定义良好的概念，取自 [6]（该文定义的是图而非超图）。在那里，作者指出，两个图的粘合是图范畴中的一个推出。本文不采用范畴论方法，因此沿用集合论定义。

注意 , 如果 I 是离散的 , 那么粘合过程可以表示为替换：

命题 1. 设 $I = D[k]$，并设 G_i、φ_i 如上。设 $G'_1 = \langle V_{G_1}、E_{G_1} \sqcup \{e_0\}、att_{G'_1}、lab_{G'_1}\rangle$，其中对于 $e \in E_{G_1}$，$att_{G'_1}(e) = att_{G_1}(e)$，$lab_{G'_1}(e) = lab_{G_1}(e)$，并且 $att_{G'_1}(e_0) = \varphi_1(1) \ldots \varphi_1(k)$（$e_0$ 的标签无关紧要）。设 $G'_2 = \langle V_{G_2}、E_{G_2}、att_{G_2}、lab_{G_2}、\varphi_2(1) \ldots \varphi_2(k)\rangle$。那么 $G_1 +_{\varphi_1,\ \varphi_2} G_2 = G'_1[e_0/G'_2]$。

该命题可由粘合和替换的定义直接推出 nt.

标签集 C 上的超图文法规则形式为 $r = (L \xleftarrow{\varphi_L} I \xrightarrow{\varphi_R} R)$，其中 L、I、R $\in \mathscr{H}(C)$ 均为零秩，且 φ_L, φ_R 是态射。若存在一个零秩超图 C 和一个态射 $\psi: I \to C$，使得 $G \cong C +_{\psi,\ \varphi_L} L$，且 $H \cong C +_{\psi,\ \varphi_R} R$ [6]（$\cong$ 表示超图同构），则称超图 G 经由 r 变换为 H。从范畴论的角度，这可以表示为一个双推出图：

$$\begin{array}{ccccc} L & \xleftarrow{\varphi_L} & I & \xrightarrow{\varphi_R} & R \\ \downarrow{\scriptstyle m} & & \downarrow{\scriptstyle \psi} & & \downarrow{\scriptstyle n} \\ G & \xleftarrow{\eta_L} & C & \xrightarrow{\eta_R} & H \end{array}$$

该变换记作 $G \Rightarrow H$，或简记为 $G \Rightarrow_r H$。

注意！下文中，我们仅考虑 I 为离散的超图规则。这并不会实质性地限制这一形式体系。

例1。考虑以下 DPO 规则 ρ：

$$\rho = \left(\begin{array}{ccc} & \{1\} & \\ l & & r \\ \{2\} & & \{3\} \end{array} \leftarrow D[3] \rightarrow \begin{array}{ccc} \{1\} & 1 & f \\ 1 & & \\ \{2\}\ 2 & t & 3\ \{3\} \end{array} \right)$$

请注意，最左侧和最右侧的超图均为零秩（圆括号中没有外部节点）；花括号中的数字表示接口超图 $D[3]$ 中节点的像（即，左侧超图中上标为 $\{1\}$ 的节点就是 $\varphi_L(1)$，其余依此类推）。

超图文法规则的定义和命题 1 表明，对于 $r = (L \xleftarrow{\varphi_L} D[k] \xrightarrow{\varphi_R} R)$，$G \overset{r}{\Rightarrow} H$ 的规则应用由逆替换 $G = C'[e_0/L'] \Leftarrow C'$ 和替换 $C' \Rightarrow C'[e_0/R'] = H$ 组成。这里，C 与超图规则应用定义中的相同；C'、e_0, L' 和 R' 的定义方式与命题 1 中 G'_1、e_0 和 G'_2 相同；特别地，$L' = \langle V_L、E_L、att_L、lab_L、\varphi_L(1) \ldots \varphi_L(k)\rangle$，$R' = \langle V_R、E_R、att_R、lab_R、\varphi_R(1) \ldots \varphi_R(k)\rangle$。

超图规则的应用可以扩展到 G 和 H 均非零秩的情形。事实上，如果存在一个超图 C'，其中有一条指定的超边（称为 e_0），使得 $C'[e_0/L'] \cong G$ 且 $C'[e_0/R'] \cong H$，那么我们就可以说 G 经由 r 变换为 H。下文将采用这个扩展定义。显然，如果 $G \Rightarrow H$，那么 $rk(G) = rk(H)$。

一个双推送补超图文法 HGr 形式为 $\langle N\ \Sigma, P, Z\rangle$，其中 N 和 Σ 是互不相交的有限字母表，分别由非终结符标签和终结符标签组成；P 是一组有限的、定义在 $N \cup \Sigma$ 上的超图文法规则，形式为 $L \xleftarrow{\varphi_L} D[k] \xrightarrow{\varphi_R} R$；Z 是一个起始超图。由 HGr 生成的语言 L(HGr) 是所有满足 $Z \Rightarrow^* H$ 的超图 $H \in \mathscr{H}(\Sigma)$ 的集合。注意，不失一般性，我们可以假设 $Z = S^\bullet$，其中 $S \in N$。

2.2 超图兰贝克演算与超图兰贝克文法 s

现在来定义超图兰贝克演算。关于其动机的详情可见[13, 15]；在下一小节中，我们还将阐述超图兰贝克演算与线性逻辑之间的关联及其动机。

我们固定一个原始类型集合 Pr，以及一个函数 rk : Pr $\to \mathbb{N}$；要求对于每个 k $\in \mathbb{N}$，都存在可数多个 p ∈Pr，使得 rk(p) = k。此外，我们固定一个可数标签集 $\$_n$，n $\in \mathbb{N}$，并令 rk($\$_n$) = n；约定这些标签不属于演算定义中涉及的任何其他集合。接下来，类型集 Tp 归纳定义如下：

1. 所有原始类型都是类型。
2. 设 N ∈Tp 是一个类型，D 是一个超图，其所有超边的标签中，除一个之外都属于 Tp，而其中一个超边的标签对于某个 d 等于 $\$_d$；再设 rk(N) = rk(D)。那么 N ÷D 也是一个类型，且 rk(N ÷D) := d。D 中标签为 $\$_d$ 的超边记作 $e_D^{\$}$。
3. 如果 M 是一个由 Tp 中的类型标记的超图，那么 ×(M) 也是一个类型，并且 rk(×(M)) :=rk(M)。

例5包含一个类型的示例（注意，× 的绑定强于 ÷）。序列式是形如 H →A 的结构，其中 H 是一个由类型标记的超图（称为该序列式的前件），而 A 是一个类型（称为后件），并且 rk(H) = rk(A)。

超图兰贝克演算 HL 推导超图序列式。HL 的唯一公理形式为 $A^{\bullet} \to A$，其中 A ∈Tp。HL 有四条推理规则：

$$\frac{H[e/N^{\bullet}] \to A \quad H_1 \to lab_D(d_1) \quad \dots \quad H_k \to lab_D(d_k)}{H\left[e/D[e_D^{\$}/(N \div D)^{\bullet}, d_1/H_1, \dots, d_k/H_k]\right] \to A}\ (\div \to) \qquad \frac{D[e_D^{\$}/F] \to N}{F \to N \div D}\ (\to \div)$$

$$\frac{H_1 \to lab_M(m_1) \quad \dots \quad H_l \to lab_M(m_l)}{M[m_1/H_1, \dots, m_l/H_l] \to \times(M)}\ (\to \times) \qquad \frac{H[e/M] \to A}{H[e/(\times(M))^{\bullet}] \to A}\ (\times \to)$$

这里 N ÷D，×(M) 是类型；e $\in E_H$；$E_D = \{e_D^{\$}, d1, \dots, d_k\}$，$E_M = \{m1, \dots, m_l\}$。在上面给出的每条规则中，横线以上的序列式称为前提，横线以下的序列式称为结论。如果一个超图序列式 H → A 可以通过应用 HL 的规则由 HL 的公理得到，则称其在 HL 中是可导出的（记作 HL ⊢H → A）。相应的规则应用序列称为推导。例6给出了一个推导示例。

超图兰贝克演算的推理规则定义为作用于超图序列式上的变换。所有规则都通过替换来定义；此外，每应用一条规则，序列的前件或后件中就会出现一个新类型。让我们仔细看看两条特定规则，即 (÷ →) 和 (× →)。规则 (÷ →) 的形式如下：给定一个序列 $H[e/N^{\bullet}] \to A$（注意，$H[e/N^{\bullet}]$ 在结构上与超图 H 相同，替换只会改变 e 的标签），以及序列 $H_i \to lab_D(d_i)$，其中 i = 1, . . . , k，我们将 H 中的 e 替换为 D，然后将标记为 $\$_d$ 的超边重新标记为类型 (N ÷D)，接着将每个 d_i 替换为对应的前件 H_i（i = 1, . . . , k）。因此，这条规则实质上由多次替换组成。与此相反，规则 (× →) 执行与替换相反的变换：如果前件中有一个超图 G[e/M] ，那么就可以将其子超图 M“压缩”为一条标记为类型 ×(M) 的单一超边 e。

注 1. 规则（× →）和（→ ÷）在 HL 中是可逆的。这意味着：

1. 如果 形式为 $H[e/(\times(M))^{\bullet}] \to A$ 的序列在 HL 中是可导出的，那么 $H[e/M] \to A$ 也可导出 .
2. 如果形如 F →N ÷D 的序列在 HL 中是可导出的，那么 $D[e_D^{\$}/F] \to N$ 也是如此。

这里的所有记号都与规则 $(\to \div)$ 和 $(\times \to)$ 中的相同。这可以利用割消去定理加以证明；该定理及其证明见于 [15,命题 1]。

HL 文法是一个元组 $\mathrm{HLGr} = \langle \Sigma, S, \triangleright \rangle$，其中 Σ 是一个字母表，并带有秩函数 $\mathrm{rk}: \Sigma \to \mathbb{N}$；$S \in \mathrm{Tp}$ 是一个特定类型；$\triangleright \subseteq \Sigma \times \mathrm{Tp}$ 是一个有限二元关系，使得 $a \triangleright T$ 蕴含 $\mathrm{rk}(a) = \mathrm{rk}(T)$。由 HL 文法 $\mathrm{HLGr} = \langle \Sigma, S, \triangleright \rangle$ 生成的语言 L(HLGr) 是所有超图 $G \in \mathscr{H}(\Sigma)$ 的集合，其中存在一个函数 $f_G: E_G \to \mathrm{Tp}$，使得：

1. 当 $e \in E_G$ 时，$\mathrm{lab}_G(e) \triangleright f_G(e)$；2. $\mathrm{HL} \vdash f_G(G) \to S$（请注意，$f_G(G)$ 是通过 f_G 对 G 进行重新标记所得的图）。

2.3 关于 HL 的一些见解

在本小节中，我们想概述 Lambek 演算、HL 与线性逻辑之间的关系，以便为超图兰贝克演算提供一些直观认识。我们可以借助 [11] 中的思想来做到这一点。在那篇论文中，作者引入了一阶乘法直觉主义线性逻辑 MILL1，并证明 Lambek 演算可以嵌入其中（其变体也可以，例如带置换的 Lambek 演算、非结合兰贝克演算等）。正如我们将要展示的，这种演算与 HL 密切相关。

本小节的内容并非理解论文主要技术结果所必需，因此，如果读者想跳过本节，我们建议直接阅读第 3 节。

Lambek 演算是一种命题逻辑，通常以 Gentzen风格呈现，也就是作为相继式演算。使用相继式演算很方便，因为与 Hilbert式公理中心演算相比，用它们更容易检验可推导性。此外，例如，如果我们考虑 Gentzen风格的经典命题演算，那么它的规则可以很好地分为两类：一类是纯粹的逻辑规则，用于描述合取、析取、蕴含或否定等逻辑运算的行为；另一类是弱化、收缩或置换等结构规则。例如，在经典逻辑中，弱化规则的形式如下：

$$\frac{\Gamma \to A}{\Gamma, B \to A}$$

其含义如下：如果能够从假设 Γ 推出 A，那么也能够从 Γ, B 推出 A。显然，这条规则并不涉及某种特定的逻辑运算，而是说明了序列本身的行为。删除部分或全部结构规则后，便得到次结构逻辑，它们在计算机科学、语言学及其他科学领域都很有用。Lambek演算就是其中之一；它既不包含弱化，也不包含收缩，甚至不包含置换。因此，序列左侧假设的顺序至关重要。这催生了一种称为积的非交换合取，以及它的两个残余运算，分别称为左除和右除。形式上，L 的类型（公式）由原子类型 Pr 通过这三种运算构成，分别记作 $\cdot$、$\backslash$ 和 $/$。序列是形如 $A1, \ldots, A_n \to A$ 的结构，其中 $n > 0$，且 A_i、A 均为类型。L 的唯一公理对每个类型 A 都具有形式 $A \to A$。共有六条规则：

$$\frac{\Gamma, A, B, \Delta \to C}{\Gamma, A \cdot B, \Delta \to C}\,(\cdot \to) \qquad \frac{\Pi \to A \quad \Psi \to B}{\Pi, \Psi \to A \cdot B}\,(\to \cdot)$$

$$\frac{\Pi \to A \quad \Gamma, B, \Delta \to C}{\Gamma, \Pi, A\backslash B, \Delta \to C}\,(\backslash \to) \qquad \frac{A, \Pi \to B}{\Pi \to A\backslash B}\,(\to \backslash) \qquad \frac{\Pi \to A \quad \Gamma, B, \Delta \to C}{\Gamma, B/A, \Pi, \Delta \to C}\,(/ \to) \qquad \frac{\Pi, A \to B}{\Pi \to B/A}\,(\to /)$$

此处大写拉丁字母表示类型，大写希腊字母表示类型序列（且 Π非空）。

例2。在 L 中，序列 p, q/r, r/s $\rightarrow$(p·q)/s 是可导出的（其中 p、q、r 和 s 是原始类型）：

$$\dfrac{\dfrac{p \to p \quad q \to q}{p, q \to p \cdot q}\ (\to \cdot) \quad \dfrac{r \to r \quad s \to s}{r/s, s \to r}\ (/ \to)}{\dfrac{p, q/r, r/s, s \to p \cdot q}{p, q/r, r/s \to (p \cdot q)/s}\ (\to /)}$$

Lambek演算有许多为语言学和逻辑学目的而设计的变体：多模态兰贝克演算、位移演算、非结合兰贝克演算等。此外，还有一些演算对这些变体进行了推广和统一，即在统一的框架下表示这些变体。其中一种推广是超图兰贝克演算。另一种是一阶直觉线性逻辑 [11], ，更准确地说，是其乘法片段 MILL1。其语言包括个体变量 x_0,x1、.....，个体常量 c_0, c1、.....，不同元数的函数符号，二元联结词 $\otimes$ 和 $\multimap$，以及量词 $\forall$ 和 $\exists$。项是将函数符号应用于变量和常量列表所得的结果。MILL1 的公式由项通过 $\otimes$、$\multimap$和量词构成，与任何一阶逻辑中的构造方式相同。例如， $\forall$x.$\exists$y.(s(x, y) $\otimes$t(x, y, c_0)) 是一个公式。序列的形式为 $\Gamma \to$ A，其中 Γ 是公式的一个多重集（即不带顺序），而 A 是一个公式。MILL1 唯一的公理模式是对所有公式 A，均有 A $\rightarrow$ A。推理规则如下：

$$\dfrac{\Gamma, A, B \to C}{\Gamma, A \otimes B \to C}\ (\otimes \to) \quad \dfrac{\Gamma \to A \quad \Delta \to B}{\Gamma, \Delta \to A \otimes B}\ (\to \otimes) \quad \dfrac{\Delta \to A \quad \Gamma, B \to C}{\Gamma, \Delta, A \multimap B \to C}\ (\multimap \to) \quad \dfrac{\Gamma, A \to B}{\Gamma \to A \multimap B}\ (\to \multimap)$$

$$\dfrac{\Gamma, A \to C}{\Gamma, \exists x.A \to C}\ (\exists \to) \quad \dfrac{\Gamma \to A[x := e]}{\Gamma \to \exists x.A}\ (\to \exists) \quad \dfrac{\Gamma, A[x := e] \to C}{\Gamma, \forall x.A \to C}\ (\forall \to) \quad \dfrac{\Gamma \to A}{\Gamma \to \forall x.A}\ (\to \forall)$$

这里，e 是我们任意选取的常量或变量（可以是 Γ 中的，也可以是 A 中的，或是一个新变量）。规则 ($\exists \to$) 和 ($\to \forall$) 要求变量 x 不在 Γ 或 C 中自由出现。

在 [11], 中，人们发现了将 L 翻译为 MILL1 的方法，使得 L 的每种类型都变换为 MILL1 中含有两个自由变量的公式：

1. $||p||^{x,\ y} = p(x, y)$; 2. $||A/B||^{x,\ y} = \forall z.||B||^{y,\ z} \multimap ||A||^{x,\ z}$; 3. $||B\backslash A||^{x,\ y} = \forall z.||B||^{z,\ x} \multimap ||A||^{z,\ y}$;
4. $||A \cdot B||^{x,\ y} = \exists z.||A||^{x,\ z} \otimes ||B||^{z,\ y}$。

最后，L 中的一个序列 $A1, \ldots, A_n \to B$ 被翻译为序列 $||A_1||^{c_0,\ c^1}, \ldots, ||A_n||^{c_{n-1},\ c^n} \to ||B||^{c_0,\ c_n}$，其中 c_0, c 1, . . . ,c_n 是互不相同的常量。证明表明，L 中的一个序列可导出，当且仅当其翻译在 MILL1 中可导出。

示例 3。例2中给出的序列类型翻译成 MILL1 如下：

1. $||p||^{x,y} = p(x, y)$; 2. $||q/r||^{x,y} = \forall z.r(y, z) \multimap q(x, z)$; 3. $||r/s||^{x,y} = \forall z.s(y, z) \multimap r(x, z)$; 4. $||(p \cdot q)/s||^{x,y} = \forall z.(s(y, z) \multimap \exists t.p(x,t) \otimes q(t, z))$.

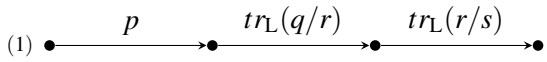
例2中的序列翻译如下：

$$^{p(c_0, c_1)}\llbracket \circ \forall z.r(c_2, z) \multimap q(c_1, z), \forall z.s(c_3, z) \multimap r(c_2, z) \ \to\ \forall z.(s(c_3, z) \multimap \exists t.p(c_0, t) \otimes q(t, z)) \quad (1)$$

验证后一个序列式可在 MILL1 中导出是一项练习。

一个重要的观察是，尽管 MILL1 是一种交换逻辑，也就是说，序列前件中的公式顺序可以任意改变，但我们仍然可以将 L 这样的非交换逻辑嵌入其中。诀窍在于，我们使用常量 c0, . . . ,c_n 来保留 L 中类型的顺序：它们固定了 L 中原始序列式的线性结构。

在 [15],中，我们证明 L 可以嵌入 HL。例如，序列 p,q/r,r/s $\rightarrow$(p ·q)/s 被翻译为以下超图序列式：



$$\rightarrow tr_{\mathrm{L}}((p\cdot q)/s) \tag{2}$$

这里，$\mathrm{tr_L}$ 是从 L 到 HL 的翻译，我们不打算在本文中进行描述。比较 L 到 MILL1 和到 HL 的嵌入后，我们观察到一些对应关系：常量 c_i (i =0, 1, 2, 3) 对应于前件（2）中图的节点；公式 $p(c_0, c_1)$ 对应于一条从节点 c_0 指向节点 c_1的 p 标记边；常量 c_0, c_2 对应于图的外部节点。我们也可以比较 HL 和 MILL1 的规则，并观察到二者高度相似。事实上，这些观察可以推广：我们声称超图兰贝克演算可以嵌入 MILL1（见下方示例）。然而，正式引入这一嵌入并证明其正确性，应留待另一篇论文处理；此外，这与本文后续的讨论无关。请记住，主要目标是帮助读者直观理解 HL 的组织方式，以及如何将指数模态添加到 HL 中。

例4。例5中的类型DPO(ρ)可以转换为以下 MILL1 公式：∀x.∀y.∀z.(f (x) $\otimes$t(x,y,z)) $\multimap$(l(x,y) $\otimes$r(x, z))。例6推导末尾出现的 HL 序列可以转换为以下 MILL1 序列：

$$f(c_0), t(c_0,c_1,c_2), p(c_1,c_2), \forall x.\forall y.\forall z.(f(x)\otimes t(x,y,z)) \multimap (l(x,y)\otimes r(x,z)) \\ \rightarrow \exists x.\exists y.\exists z.l(x,y)\otimes r(x,z)\otimes p(y,z)$$

考虑到这一点，我们希望继续通过指数来扩展 HL。

3 超图乘法-指数 Lambek 演算 **us**

在本节中，我们展示一种通过指数模态 ! 扩展 HL 的方法，由此得到超图乘法指数 Lambek 演算 $\mathrm{HMEL_0}$ （稍后我们会解释这个下标 0）。完成后，我们将展示如何把任意 DPO 文法转换为等价的 $\mathrm{HMEL_0}$-文法。该转换过程与 [4] 中介绍的方法类似：在那里，每个无约束的乔姆斯基层次文法都会被转换为乘法-指数 Lambek 演算上的等价文法。

在第 2.3 节中，我们介绍了直觉线性逻辑片段的规则。现在，让我们来看完整的逻辑 ILL1，它包含指数。ILL1 的公式由 $\otimes$、 $\multimap$、1、&、$\oplus$、! 以及量词 $\forall$ 和 $\exists$ 构成。ILL1 的规则包括 MILL1 的规则，以及定义新联结词的规则。让我们重点关注 ! 模态的规则：

$$\frac{\Gamma, A \rightarrow B}{\Gamma, !A \rightarrow B}\ (!\rightarrow) \qquad \frac{!A_1,\ldots,!A_n \rightarrow B}{!A_1,\ldots,!A_n \rightarrow !B}\ (\rightarrow !) \qquad \frac{\Gamma \rightarrow B}{\Gamma, !A \rightarrow B}\ (\mathrm{w}) \qquad \frac{\Gamma, !A, !A \rightarrow B}{\Gamma, !A \rightarrow B}\ (\mathrm{c})$$

我们的目标是将这些规则迁移到 HL 中。若能不受限制地将指数模态引入 HL，也就是说，对于每一种类型 A 都能考虑类型 !A，那将十分理想。然而，如果我们希望所得演算具有良好的性质，即满足切割

消去定理，那么这会带来一些困难。遗憾的是，目前我们还没有为超图演算中的指数提出一种通用处理方法，使得该定理能够成立，因此这仍是一个开放问题。不过，如果我们作出限制，仅当 A 的秩为 0（$\mathrm{rk(A)} = 0$）时才允许考虑类型 !A，那么这样的演算便可以得到良好定义；尤其是，它满足割消去定理。

我们建议为 HL 添加以下指数模态规则：

$$\frac{H + A^\bullet \to B}{H + (!A)^\bullet \to B}\ (!\to) \qquad \frac{\sum_{i=1}^n (!A_i)^\bullet \to A}{\sum_{i=1}^n (!A_i)^\bullet \to !A}\ (\to !) \qquad \frac{H \to B}{H + (!A)^\bullet \to B}\ (\mathrm{w}) \qquad \frac{H + 2\cdot(!A)^\bullet \to B}{H + (!A)^\bullet \to B}\ (\mathrm{c}) \tag{3}$$

这里，$\mathrm{rk}(!A_i) = \mathrm{rk}(A_i) = \mathrm{rk}(!A) = \mathrm{rk}(A) - 0$。求和符号表示多个不交并。显然，这些规则推广了此前针对 ILL1 考虑的字符串对应规则。注意，若 $\mathrm{rk}(X) = 0$，则 $X^\bullet$ 是一个零秩超边，作为前件超图中的一个独立分量“浮动”。

定义1. 超图乘法-指数 Lambek 演算 HMEL_0 的定义与 HL 类似，但在类型 Tp 的定义中，我们还规定：若 A 是一个满足 $\mathrm{rk(A)} = 0$ 的类型，则 !A 也是一个类型，并且满足 $\mathrm{rk}(!A) = 0$。我们将 (3) 中的规则添加到 HL 的规则中。HMEL_0 文法的定义方式与 HL 文法相同，但其基础是 HMEL_0。

下标0对应于仅允许在秩为0的类型上添加 ! 的限制。正如我们提到的，这使我们能够证明以下定理：

定理1. 如果 $\mathrm{HMEL}_0 \vdash H \to A$ 且 $\mathrm{HMEL}_0 \vdash G[e_0/A^\bullet] \to B$，那么 $\mathrm{HMEL}_0 \vdash G[e_0/H] \to B$。

换言之，该定理表明，我们可以将下面介绍的切割规则添加到规则列表中，而不会扩大可导出的序列式集合：

$$\frac{H \to A \quad G[e_0/A^\bullet] \to B}{G[e_0/H] \to B}\ (\mathrm{cut})$$

证明。我们证明，下面介绍的切割规则和混合规则都可以添加到 HMEL_0 中，而不影响可导出的序列式集合：

$$\frac{H \to !C \quad G' + (!C)\cdot n \to B}{G' + H \to B}\ (\mathrm{mix})$$

这里 H 是零秩的。此类规则通常用于证明包含弱化（w）和收缩（c）的线性逻辑变体的割消去定理。事实上，整个证明与 [9], 中的证明相似，这对于带指数的次结构逻辑（如 HMEL_0）而言是典型的。为保持统一性，我们将 !C 记为 A，并将 $G' + (!C)\cdot n$ 和 $G[e_0/A^\bullet]$ 都记为 F。

证明通过嵌套归纳完成：外层归纳针对 A 的大小（计数方式为构造 A 时所用原始类型以及符号 $\times$、$\div$ 和 ! 的总数），内层归纳针对 $H \to A$ 和 $F \to B$ 的推导长度之和。我们需要根据 $H \to A$ 或 $F \to B$ 的推导中最后一条规则，考虑若干情形。一个重要概念是主类型：给定某个序列的推导，主类型就是推导最后一步之后序列中出现的类型。例如，在例2中，主类型是 $(p\cdot q)/s$；在例6中，主类型是 DPO(ρ)。

第一组情形是 $H \to A$ 或 $F \to B$ 为公理；此时定理的陈述显然成立。

第二组是指：后件 $H \to A$ 中的 A 在 $H \to A$ 的推导中不是主公式。此时，最后一条规则以某种方式涉及 H，但不会影响 A。因此，我们可以重复相同的

步骤，不过是在 G 中用 H 替换 e，然后应用归纳假设。更正式地说，设 H $\to$A 的推导的最后一步如下：

$$\frac{H' \to A \quad \mathscr{P}}{H \to A}$$

这里 $\mathscr{P}$ 还有一些额外的前提。然后，我们对 H′ $\to$ A 和 F $\to$B 应用归纳假设，从而得出 G[e_0/H′] $\to$ B（对于混合规则则为 G′ +H′ $\to$ B）是可导出的。最后，我们执行以下步骤：

$$\frac{G[e_0/H'] \to B \quad \mathscr{P}}{G[e_0/H] \to B}$$

这完成 了这一组情形的证明。从现在起，我们假设 A 是 H $\to$A 中的主公式 .

第三组情形是：A 的指定出现（或者，如果考虑（混合），则 A =!C 的全部 n 个指定出现）在 F $\to$B 的推导中不是主公式，并且其推导中的最后一条规则不是（$\to$! ）。然后，可以采用与第二组情形相似的推理。

第四组情形是：A 在 H $\to$ A 和 F $\to$B 中都是主公式，但 A 不是 !C 的形式（这只可能出现在切割规则中）。然后，推理与 HL 中的相同，见 [15, 附录 A]。

在其余情形中，A =!C（因此切割规则变为混合规则的一个实例）在 H $\to$ A 和 F $\to$B 中都是主公式；我们还需要考虑 F $\to$ B 的推导中最后一条规则为 ($\to$!) 的情形。在所有这些情形中，H $\to$ A 的推导中最后一条规则都是 ($\to$!)，因此，序列 H $\to$C 可导出，且 H $= \sum_{i=1}^{n}(!A_i)^\bullet$，其中某个 A_i 满足 rk(A_i) $= 0$。根据 F $\to$B 的推导中所应用的最后一条规则，可能出现以下情形：

情形 I。最后一条规则应用是 (! $\to$)。那么它必定具有如下形式：

$$\frac{G' + (!C) \cdot (n-1) + C^\bullet \to B}{G' + (!C) \cdot n \to B}\ (! \to)$$

利用归纳假设，我们对 H $\to$A 和 G′ + A $\cdot$(n − 1) +C• $\to$B 应用混合规则，得出 G′ +H +C• $\to$ B 可导出。然后，我们再次对序列 H $\to$ C 和 G′ + H +C• $\to$ B 应用归纳假设（注意，C 的大小小于 A，因此第一个归纳参数减小），得到序列 G′ +H +H $\to$ B。最后，应用收缩规则 (c) 共 n 次，从而将两个 H 收缩为一个 H。

情形 II。最后一条规则应用是 (w)。那么它必定具有以下形式：

$$\frac{G' + (!C) \cdot (n-1) \to B}{G' + (!C) \cdot n \to B}\ (! \to)$$

接下来只需将归纳假设应用于 H $\to$A 和 G′ + (!C) $\cdot$ (n − 1) $\to$B。

情况 III。最后一条规则应用是 (c)。推理与前一种情况类似 e.

情况 IV。最后一条规则应用是 ($\to$!)。下文中，G′ $= \sum_{i=1}^{m}(!D_i)^\bullet$ 对某些类型 D_i、B =!D 和 F $\to$ D 是可导出的。让我们将归纳假设应用于 H $\to$A 和 F $\to$D，从而得出 $\sum_{i=1}^{m}(!B_i)^\bullet + \sum_{i=1}^{n}(!A_i)^\bullet \to$ D 是可导出的。最后，我们应用规则 ($\to$!)，得到所需的 G′ +H $\to$ B。 □

作为割消去定理的推论，我们注意到注 1 对 $HMEL_0$ 也成立。现在，我们的目标是证明，基于新演算的 $HMEL_0$-文法至少与 DPO 文法具有同等的表达能力。这可以通过类似的方法证明：基于乘法-指数 Lambek演算的文法至少与无约束 Chomsky 文法具有同等的表达能力。回想一下

乔姆斯基层次文法中的规则形式为 $\alpha \Rightarrow \beta$，其中 α、β 是由非终结符号和终结符号组成的任意两个字符串。[4]中的证明表明，可以将形式为 $A_1 \dots A_n \Rightarrow B_1 \dots B_m$（其中 A_i、B_j是非终结符号）的每条规则转换为以下类型:

$$[\tau(A_1 \dots A_n \Rightarrow B_1 \dots B_m)] := (A_1 \cdot \dots \cdot A_n)/(B_1 \cdot \dots \cdot B_m).$$

Makoto Kanazawa 在 [4]中证明，对于给定的由上述形式规则组成的有限集合 $R = \{r1, \dots, r_k\}$，非终结符号串 w 可由非终结符号 S 使用 R 中的规则导出，当且仅当 MELC $\vdash ![\tau(r_1)], \dots, ![\tau(r_k)], w \to S$，其中所有非终结符号均视为原始类型，且 w 中的所有符号均以逗号分隔。例如，规则 $S \Rightarrow$SSA 允许从 S 生成 SSASA，因此 MELC $\vdash !(S/(S \cdot S \cdot A)), S, S, A, S, A \to S$。MELC 是乘法-指数 Lambek 演算的缩写。

同样的思路也可以应用于 DPO 规则。首先，我们需要对 DPO 超图文法稍作扩展。

构造 1。给定一个 DPO 文法 $\langle N\ \Sigma, P, S^\bullet \rangle$，我们将其转换为一个文法 $\langle N', \Sigma, P', S^\bullet \rangle$，称之为规范化文法，具体如下。对于每个 $a \in \Sigma$，我们引入一个新的非终结标签 T_a，使得 $rk(T_a) = rk(a)$；令 $N' = N \sqcup \{T_a \mid a \in \Sigma\}$。接着，对于每个 $r = (L \xleftarrow{\varphi_L} D[k] \xrightarrow{\varphi_R} R) \in P$，我们将 L 和 R 中的每个终结标签 a 替换为 T_a。我们将这些新规则称为非终结符规则，并将其集合记作 P_N。最后，我们添加允许将 T_a 替换为 a 的规则，即形如 $(T_a^\circ \xleftarrow{\varphi_L} D[k] \xrightarrow{\varphi_R} a^\circ)$ 的规则，其中 $rk(a) = k$，$\varphi_L(i) = \varphi_R(i) = i$，$i = 1, \dots, k$（这里使用定义 D[k] 和 S° 中对节点的记法）。这些规则称为终结规则，记作 P_T。最后，$P' = P_N \cup P_T$。

显然，规范化文法生成的语言与原文法相同。下文我们只考虑规范化文法。

非终结符规则可以转换为 HL 的相应类型，如下所示:

构造 2. 将规范化文法的非终结符标签视为原始类型（具有相同的秩函数）。若 $r = (L \xleftarrow{\varphi_L} D[k] \xrightarrow{\varphi_R} R)$ 是一条非终结符规则，则 $DPO(r) := \times\left(\widehat{\ }L\right) \div \left(\widehat{\ }R + \$_0^\bullet\right)\widehat{\ }$

其中 $L = \langle V_L, E_L, att_L, lab_L, \varphi_L(1) \dots \varphi_L(k) \rangle$; $R = \langle V_R, E_R, att_R, lab_R, \varphi_R(1) \dots \varphi_R(k) \rangle$。注意，$\$_0^\bullet$ 是一个秩为 0 的独立超边，并且 $rk(DPO(r)) = 0$。

例5. 例1中的非终结符规则 ρ 转换为以下类型DPO(ρ):

$$DPO(\rho) = \times \left(\begin{array}{ccc} & l \quad r & \\ & (1) & \\ (2) & & (3) \end{array} \right) \div \left(\begin{array}{ccc} & (1)\ 1\ \boxed{f} & \\ & 1 & \\ (2)\ 2\ \boxed{t}\ 3 & & (3) \end{array} \quad \boxed{\$_0} \right)$$

r 与 DPO(r) 之间的主要联系见

引理1. 设 Y,Y ′为两个超图，且 $Y \Rightarrow Y'$对于 $r \in P_N$。若对于任意类型 A，HL $\vdash Y \to A$，则 r HL $\vdash Y' + DPO(r)^\bullet \to A$ 也成立。

引理2. 在规范化文法 $\Rightarrow^* Y'$中，$\langle N\ \Sigma, P, S^\bullet \rangle$ 通过非终结符规则，其中 Y,Y'是超图。若 HL $\vdash Y \to$对于任意类型 A，HL $\vdash Y' + \sum k_r \cdot DPO(r)^\bullet \to A$，其中某个 $k_r \in \mathbb{N}$。$r \in P_N$

主要引理3。设 Y ′为一个超图；设 $\langle N\ \Sigma, P, S^\bullet \rangle$ 为一个规范化文法，其非终结符规则集合为 P_N；设 X 为一个非终结符号。当且仅当对于某些 $b_r, k_r \in \mathbb{N}$，存在 $r \in P_N$ $r \in P_N$，使得 HL $\vdash Y' + \sum b_r \cdot (!DPO(r))^\bullet + \sum k_r \cdot (DPO(r))^\bullet \to X$ 时，$X^\bullet \Rightarrow^* Y'$可使用 P_N中的规则进行推导。

引理1通过直接应用（$\times\to$），然后应用（$\div\to$）即可证明序列 $Y\ \to A$（下文给出了一个代表性示例）。引理2直接由引理1推出。注意，其中的 k_r 是推导 $Y\ \Rightarrow^* Y'$中应用 r 的次数。

因此，可以在 HL 中重构 DPO 推导，但 r 的每次规则应用都会留下痕迹，即前件中有一条标记为 DPO(r) 的悬浮超边。借助指数模态，我们可以将这些超边合并为一条。

例6。以下推导说明了引理1：

$$\dfrac{\dfrac{\dfrac{l^\bullet\to l\quad r^\bullet\to r\quad p^\bullet\to p}{\begin{array}{c}l\quad r\\ \to\times\\ p\end{array}\ \to\times\left(\begin{array}{c}l\quad r\\ p\end{array}\right)}(\to\times)\qquad t^\bullet\to t\quad f^\bullet\to f}{\begin{array}{c}1\\ 2\ \boxed{\times(\widehat{L})}\ 3\\ p\end{array}\ \to\times\left(\begin{array}{c}l\quad r\\ p\end{array}\right)}(\times\to)}{\begin{array}{c}1\quad\boxed{f}\\ 1\quad\boxed{t}\\ 2\qquad 3\\ p\end{array}\quad\boxed{\mathrm{DPO}(\rho)}\quad\to\quad\times\left(\begin{array}{c}l\quad r\\ p\end{array}\right)}(\div\to)$$

这里，序列 $Y\ \to A$ 等于 $l\ r\ p\ \to\times\left(\begin{array}{c}\\ l\ r\ p\\ \triangle\quad\triangle\end{array}\right)$，规则 r 等于 ρ，见例1。

剩下要证明引理3：

证明（引理3）。「仅当」方向可由引理2直接得出：若取 $Y\ =X^\bullet$、$A\ =X$，则 $Y\ \to A$ 是公理 $X^\bullet\to X$，因此对于某个 k_r，$Y'+\ \sum k_r\cdot \mathrm{DPO(r)}^\bullet\to A$ 是可导出的。$r\in P_N$最后，使用规则 (! →)、(w) 和 (c)，可以由此导出序列 $Y'+\ \sum(!\mathrm{DPO(r)})^\bullet\to X$。$r\in P_N$

“若”方向通过对 HMEL_0 中 $Y'+\ \sum b_r\cdot(!\mathrm{DPO(r)})^\bullet+r\in P_N\sum k_r\cdot(\mathrm{DPO(r)})^\bullet\to X$ 的推导长度进行归纳来证明。为简洁起见，我们将和 $\sum b_r\cdot(!\mathrm{DPO(r)})^\bullet+r\in P_N\sum k_r\cdot(\mathrm{DPO(r)})^\bullet$ 记作 $\sigma\{b_r;k_r\}$。$r\in P_N$

基础情形显然成立。下面证明归纳步骤。证明通过考察序列推导中应用的最后一条规则来完成。

情形 1。应用的最后一条规则是 $(\div\to)$。在这种情况下，它的应用必定具有以下形式

$$\frac{Y^0\left[e_0/(\times(\widehat{L}))^\bullet\right]+\sigma\{b_r^0,k_r^0\}\to X\quad Y^1+\sigma\{b_r^1,k_r^1\}\to lab_{\widehat{R}}(e_1)\quad\ldots\quad Y^l+\sigma\{b_r^l,k_r^l\}\to lab_{\widehat{R}}(e_l)}{Y'+\sigma\{b_r',k_r'\}+(\mathrm{DPO}(r_0))^\bullet\to X}$$

这里，$\mathrm{DPO(r_0)}\ =\times(\widehat{\ }\mathrm{L})\ \div(\ \widehat{\ }\mathrm{R}\ +\$_0^\bullet);\ E\ \frown_R\ =\{e1,\ldots,e_l\};\ b_r'=b_r$对于所有 $r\ \in P_N$，$k_r'=k_r$对于 $r\ \neq r_0$和 $k_{r_0}'=k_{r_0}-1$；对于所有 $r\ \in P_N$，都有 $b_r'=b_r^0+\ldots+b_r^l$，$k_r'=k_r^0+\ldots+k_r^l$；并且

$$Y'=Y^0\left[e_0/\widehat{R}\left[e_1/Y^1,\ldots,e_l/Y^l\right]\right].$$

这只是给定序列中可应用的（$\div \to$）规则应用的一般显式形式。注 1 意味着序列 $Y^0\left[e_0/\widehat{\ }L\right]+\sigma\{b_r^0,\ k_r^0\}\to X$ 也是可导出的。应用归纳假设，我们得到 $X^\bullet \Rightarrow^* Y^0\left[e_0/\widehat{\ }L\right]$。接着，$Y^0\left[e_0/\widehat{\ }L\right] \underset{r_0}{\Rightarrow} Y^0\left[e_0/\widehat{\ }R\right]$。最后，让我们对每个 $i=1,\dots,l$ 将归纳假设应用于序列 $Y_i+\sigma\{b_r^i, k_r^i\}\to \mathrm{lab}_{\hat{R}}(e_i)$，因此可得 $\mathrm{lab}_{\hat{R}}(e_i)^\bullet \Rightarrow^* Y_i$。现在，只需从超图 $Y^0\left[e_0/\widehat{\ }R\right]$ 开始，依次在其中重做每个推导 $\mathrm{lab}_{\hat{R}}(e_i)^\bullet \Rightarrow^* Y_i$（对于 $i=1,\dots,l$），从而将 R 中的每条超边 e_i 替换为 Y_i。因此，我们得到

$$X^\bullet \Rightarrow^* Y^0\left[e_0/\widehat{L}\right] \Rightarrow Y^0\left[e_0/\widehat{R}\right] \Rightarrow^* Y^0\left[e_0/\widehat{R}\left[e_1/Y^1,\dots,e_l/Y^l\right]\right] = Y'.$$

情形 2。最后应用的规则或者是 (i) 规则 $(!\to)$，或者是 (ii) 规则 (w)，或者是 (iii) 规则 (c)。假设该规则应用的主要类型是 $!\mathrm{DPO}(r_0)$，其中 $r_0 \in P_N$。那么它必定具有如下形式

$$\frac{Y'+\sigma\{b_r', k_r'\}\to X}{Y'+\sigma\{b_r, k_r\}\to X}$$

这里，对于 $r \neq r_0$，有 $b_r'=b_r$、$k_r'=k_r$；在情形 (i) 中，$b_{r_0}'=b_{r_0}-1$、$k_{r_0}'=k_{r_0}+1$；在情形 (ii) 中，$b_{r_0}'=b_{r_0}-1$、$k_{r_0}'=k_{r_0}$；在情形 (i) 中，$b_{r_0}'=b_{r_0}+1$、$k_{r_0}'=k_{r_0}$。归纳假设完成了全部三个子情形下的证明。□

现在，我们介绍一个与给定 DPO 文法相对应的 HMEL_0-文法。

构造 3。设 $\mathrm{HGr}=\langle N, \Sigma, P, S^\bullet\rangle$ 是一个规范化 DPO 文法。那么 $\mathrm{LG}(\mathrm{HGr})=\langle \Sigma, S', \triangleright\rangle$，其中 $\triangleright$ 由形如 $a \triangleright T_a$ 的对组成，并且 $S'=S \div \left(\sum_{r\in P_N} (!\mathrm{DPO}(r))^\bullet + \$_0^\bullet\right)$。

在此，我们将指数作用于每个类型 DPO(r)（其中 $r \in P_N$），并将结果存入 S'。

事实证明，这一构造是对 [4] 中构造的直接推广。利用它，我们可以证明以下引理：

引理 4. 若 HGr 是规范化 DPO 文法，则 $L(\mathrm{LG}(\mathrm{HGr})) = L(\mathrm{HGr})$。

证明. 超图 $H \in \mathscr{H}(\Sigma)$ 属于 $L(\mathrm{LG}(\mathrm{HGr}))$，当且仅当 $\mathrm{HMEL}_0 \vdash t(H) \to S'$，其中 t(H) 是 H 的重新标记，满足 $t(e)=T_{lab_H(e)}$。根据注 1，对于 HMEL_0，此序列与序列 $t(H)+\sum (!\mathrm{DPO}(r))^\bullet \to S$ 等价可导出。根据引理 2 和引理 3，当且仅当 $S^\bullet \Rightarrow^* t(H)$ 可通过 HGr 的非终结符规则导出时，$r\in P_N$ 才成立。显然，这等价于 $H \in L(\mathrm{HGr})$。

□

最后，我们得出以下结果：

定理2. HMEL_0文法至少与DPO 文法具有同等的表达能力。

4 具有线性限制的 DPO 文法与 HL 文法

上述构造高度依赖于指数模态，它使我们能够将前件中同一类型的多个副本压缩为一个副本。在没有 ! 模态的 HL 中，构造 3 不再适用。此外，它显然必然失效：HL 文法生成的语言是可判定的，甚至属于 NP，而 DPO 文法生成

至少所有递归可枚举字符串语言，因此这两类文法生成的语言并不相同。然而，构造 2 仍然有效，因为它只使用 $\times$ 和 $\div$。这有助于将 HL 文法与具有某种限制的 DPO 文法联系起来，该限制将在下文中形式化定义。

定义 2. 给定一个 DPO 超图文法 $\mathrm{HGr} = \langle \mathrm{N}, \Sigma, \mathrm{P}, \mathrm{S}^\bullet \rangle$，令 $\mathrm{L}_c(\mathrm{HGr})$ 由所有超图 $\mathrm{H} \in \mathrm{L}(\mathrm{HGr})$ 组成，使得存在一个推导 $\mathrm{S}^\bullet \Rightarrow^* \mathrm{H}$，且其步骤数不超过 $\mathrm{c} \cdot |\mathrm{E}_H|$。

构造 4. 设 $\mathrm{HGr} = \langle \mathrm{N}, \Sigma, \mathrm{P}, \mathrm{S}^\bullet \rangle$ 是一个规范化文法；设 $\mathrm{c} \in \mathbb{N}$。则我们构造 HL 文法 $\mathrm{LG}_c(\mathrm{HGr}) = \langle \Sigma, \mathrm{S}, \triangleright \rangle$，其中 $\triangleright$ 恰好包含以下各对：

$$a \triangleright \times \left(T_a^\bullet + \sum_{r \in P_N} k_r \cdot \mathrm{DPO}(r)^\bullet \right) \text{ for } k_r \geq 0, \sum_{r \in P_N} k_r \leq c.$$

注意，由于满足上述要求的 $\mathrm{k}_r \in \mathbb{N}$ 只有有限多个，因此 $\triangleright$ 是一个有限关系。例 7. 考虑一个 DPO 文法 $\mathrm{HGr} = \langle \mathrm{N}, \Sigma, \mathrm{P}, \mathrm{S}^\bullet \rangle$，其中 $\mathrm{N} = \{\mathrm{S}\}$、$\Sigma = \{\mathrm{a}\}$，且 $\mathrm{P} = \{\mathrm{r}_1, \mathrm{r}_2, \mathrm{r}_3\}$：

1. $r_1 = \left(\boxed{S} \leftarrow \mathrm{D}[0] \rightarrow \mathrm{D}[0] \right)$；
2. $\mathrm{r}_2 = \left(\mathrm{S} \quad \boxed{} \leftarrow \mathrm{D}[0] \rightarrow \boxed{S} \quad \bullet \right)$；
3. $\mathrm{r}_3 = \left({}_{\{1\}}\bullet \quad \bullet_{\{2\}} \leftarrow \mathrm{D}[2] \rightarrow {}_{\{1\}}\bullet \xrightarrow{a} \bullet_{\{2\}} \right)$。

不难看出，它生成了所有图（其中的边都有两个附着节点）：规则 r_2 生成节点，而 r_3 生成边。举例来说，考虑以下推导：

$$\boxed{S} \underset{r_2}{\Rightarrow} \boxed{S}\bullet \underset{r_2}{\Rightarrow} \boxed{S}\bullet\bullet \underset{r_1}{\Rightarrow} \bullet\quad\bullet \underset{r_3}{\Rightarrow} \bullet \xrightarrow{a} \bullet \underset{r_3}{\Rightarrow} \bullet \xrightarrow{a} \bullet \circlearrowleft a \underset{r_3}{\Rightarrow} a \circlearrowright \bullet \xrightarrow{a} \bullet \circlearrowleft a \tag{4}$$

注意，HGr 尚未规范化；使用构造 1，我们将 r_3 替换为以下两条规则：

1. $r_3' = \left({}_{\{1\}}\bullet \quad \bullet_{\{2\}} \leftarrow \mathrm{D}[2] \rightarrow {}_{\{1\}}\bullet \xrightarrow{T_a} \bullet_{\{2\}} \right)$；
2. $r_3'' = \left({}_{\{1\}}\bullet \xrightarrow{T_a} \bullet_{\{2\}} \leftarrow \mathrm{D}[2] \rightarrow {}_{\{1\}}\bullet \xrightarrow{a} \bullet_{\{2\}} \right)$.

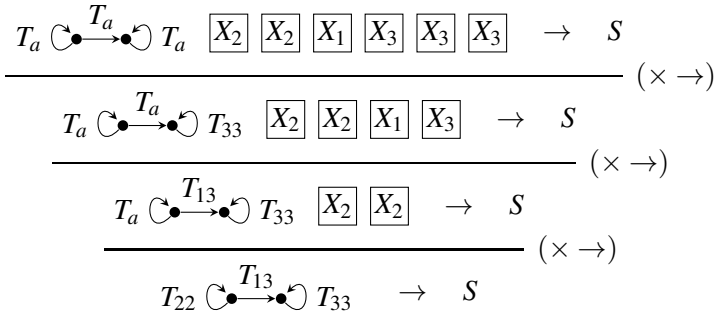
将这个新的规范化文法记为 HGr′。然后，我们使用构造 2 将其非终结符规则转换为类型：

1. $X_1 = \mathrm{DPO}(r_1) = \times \left(\boxed{S} \right) \div \left(\boxed{\$_0} \right)$；
2. $X_2 = \mathrm{DPO}(r_2) = \times \left(\boxed{S} \right) \div \left(\boxed{S} \bullet \boxed{\$_0} \right)$；
3. $X_3 = \mathrm{DPO}(r_3') = \times \left({}_{(1)}\bullet \quad \bullet_{(2)} \right) \div \left({}_{(1)}\bullet \xrightarrow{T_a} \bullet_{(2)} \quad \boxed{\$_0} \right)$；

最后，我们根据构造 4 引入一个 HL 文法 $\mathrm{LG}_2(\mathrm{HGr}') = \langle \Sigma, \mathrm{S}, \triangleright \rangle$。二元关系 $\triangleright$ 由以下 13 个有序对组成（实际上只有 10 个不同的有序对），其中 $\mathrm{i}, \mathrm{j} \in \{1, 2, 3\}$：

- $a \triangleright T = \times \left({}_{(1)}\bullet \xrightarrow{T_a} \bullet_{(2)} \right)$；
- $a \triangleright T_i = \times \left({}_{(1)}\bullet \xrightarrow{T_a} \bullet_{(2)} \quad \boxed{X_i} \right)$；
- $a \triangleright T_{ij} = \times \left({}_{(1)}\bullet \xrightarrow{T_a} \bullet_{(2)} \quad \boxed{X_i} \quad \boxed{X_j} \right)$.

回想一下，要检查一个超图是否属于 L(LG$_2$(HGr′))，我们需要：1）通过 ▷ 将其超边的标签替换为与之对应的类型；2）构造一个序列，其中前件为替换后得到的超图，后件为 S；3）导出该序列。举例来说，我们来检查 H = a aa 是否属于 L(LG$_2$(HGr′))。我们将每个标签 a 替换为类型 T、T$_i$ 或 T$_i$T$_j$（具体如下：$T_{22}\,T_{33}T^{13}$（将类型的下标与（4）中应用的规则数量进行比较）。然后，只需检查 $T_{22}\,T_{33}T^{13}$ →S 是否可导出：



上述推导中的最上方序列是可导出的，这一点可由引理2的证明得出。这证明了 H ∈L(LG$_2$(HGr′))。

上述示例说明了以下定理，这是本文的主要定理：

定理 3。对于每个 DPO 文法 HGr 和每个 c，都存在一个生成 L$_c$(HGr) 的 HL 文法。

引理 5。如果 HGr 是规范化 DPO 文法且 $1 \le c \in \mathbb{N}$，则 L(LG$_{c-1}$(HGr)) = L$_c$(HGr)。

包含关系 L(LG$_{c-1}$(HGr)) ⊇ L$_c$(HGr) 的证明使用了引理2，其方式与示例 7 中相同。另一个包含关系的证明使用了引理 3，并注意到这里前件中的 DPO(r) 标记超边数量严格对应于 HGr 中的规则应用数量（因为我们没有带指数的类型）。

构造 4 和定理 5 的思路是，在对应于终结符号的每种类型中存储带 DPO(r) 标签的超边（因为我们无法像构造 3 那样将它们存储在后件中）。接着，对于 G ∈L(LG$_c$(HGr))，将 G 中的每个符号 a 替换为类型 $\times\left(T \bullet_a + \Sigma_{r\in P_N} k_r \cdot \mathrm{DPO(r)}^{\bullet}\right)$for some k_r and start deriving a corresponding sequent, these hyperedges eventually appear in the an- tecedent where they play their role shown in Lemma 2. The total number of these hyperedges, however, is limited by the number of hyperedges in G, hence the language L$_c$(HGr) is generated instead of L(HGr).

定理 5 表明，HL 文法足以生成由 DPO 文法生成的语言中的超图，并且其推导步数受超边数量的线性函数所界定。对于某个 DPO 文法 HGr，可能有 L(HGr) =L$_c$(HGr)，其中 $c \in \mathbb{N}$；事实上，我们声称，对于每个 HL 文法 HLGr = ⟨Σ, S ▷⟩ 都存在一个 DPO 文法 HGr 和 $c \in \mathbb{N}$，使得 L(HLGr) = L$_c$(HGr) = L(HGr)，尽管我们在此不予证明（这应留待另一篇论文讨论）。然而，一般而言，L(HGr) ≠ L$_c$(HGr)（例如在例 7 中，L$_{k+1}$(HGr) 仅包含满足 $|V_G| < k \cdot |E_G|$ 的图 G）。

5 带合取 Kleene 星的 HL

在第 3 节中，我们为 HL 加入了指数模态，这帮助我们证明了定理 4。另一种表现类似的运算是 Kleene星号。许多研究者都研究了带 Kleene星号的 Lambek演算，尤其是 [7]。带有交集、并集和 Kleene星号扩展的 L 称为无穷动作逻辑 [12], ；在代数框架下，它也称为动作代数逻辑。请注意，Kleene星号可以用迁移系统中的动作来描述：若 A 是一个动作类，那么 A^* 表示重复若干次的 A 中的动作 [7]。这种理解与我们在构造 3 中采用的方式非常接近，因为我们的目标是能够在 HL 中任意多次应用编码后的 DPO 规则。因此，现在让我们来看看带单位元和 Kleene星号的 Lambek演算 $\mathbf{L}^*_{\mathbf{1}\omega}$，更具体地说，看看 Kleene星号的规则 [7]:

$$\frac{\Pi \to A^n}{\Pi \to A^*}\ (\to^*),\ n \geq 0 \qquad\qquad \frac{(\Gamma, A^n, \Delta \to B)_{n=0}^{\infty}}{\Gamma, A^*, \Delta \to B}\ (^*\to)_\omega$$

这里 $A^0 := \mathbf{1}$、$A^{n+1} := A^n \cdot A$；1 是积的单位元，满足公理 $A \cdot \mathbf{1} \leftrightarrow A$。请注意，规则（$\to^*$）实际上是每个 $n \geq 0$ 对应的一组可数条规则；相反，（$^*\to$）ω 是一条具有可数多个前提的单一规则。我们来明确一下在此演算中可导出的概念：$\mathbf{L}^*_{\mathbf{1}\omega}$ 中可导出的序列式集合，是包含 $\mathbf{L}^*_{\mathbf{1}\omega}$ 的所有公理（即所有形如 $A \to A$ 的序列式以及序列式$\to \mathbf{1}$）的最小集合 S，并且该集合在应用所有推理规则时封闭（也就是说，对于某条规则，如果横线上方的所有序列式都属于 S，那么横线下方的序列式也必须属于 S）。换言之，$\mathbf{L}^*_{\mathbf{1}\omega}$中的推导仍是一系列规则应用，只是其规模现在可以是可数的，但不包含无限长的分支。

遗憾的是，Kleene 星号的规则并不理想：它允许在序列的后件中无限复制类型（也就是说，如果后件中有 n 个 A 的副本，我们就可以将它们合并为一个类型 A^*），但不允许在前件中这样做。这促使我们考虑一种行为与之相反的运算：

$$\frac{(\Pi \to A^n)_{n=0}^{\infty}}{\Pi \to {}^*A}\ (\to^*)_\omega \qquad\qquad \frac{\Gamma, A^n, \Delta \to B}{\Gamma, {}^*A, \Delta \to B}\ (^*\to),\ n \geq 0$$

我们将运算 $^*A$ 称为合取 Kleene 星。在代数上，它可以在完备剩余格中利用无穷合取定义为 $^*a = \bigwedge a^n = \inf\{a^n \mid n \in \mathbb{N}\}$（因此我们称其为合取。$=0$

式）。请注意，这种运算的语言语义并不理想：如果 L 是一种语言（$L \subseteq \Sigma^*$），1 等于$\{\Lambda\}$，并且语言的乘法表示其词的逐对连接，而合取表示交集，那么 $^*L = \{\Lambda\}$ 当且仅当 $\Lambda \in L$ 且 $^*L = $，否则为 /0。

遗憾的是，我们在文献中几乎没有找到关于这一运算的内容。不过， [10] 在存储算子的语境下提到了它。MTL代数中的存储算子 I 的作用如下：I(a) 是所有不大于 a 的幂等元中最大的一个（在带存储的 MTL代数中，它必然存在）。可以证明，如果 $\inf\{a^n \mid n \in \mathbb{N}\}$ 存在，那么它等于 I(a)，因此这两个算子非常接近。在 [10],中提到，存储算子与 Girard指数 ! 有许多相似之处。

下面说明如何将合取 Kleene 星及其推理规则推广到超图。这里产生了一个问题：应如何理解一个类型的迭代，即 A^n? 在字符串情形中，这意味着将一个类型重复 n 次，并将各个副本依次写出，用 $\cdot$连接起来。我们需要将这一迭代过程扩展到超图。我们提出如下的一般定义：

定义 3. 秩为 k 的模板 T 是一个超图 $T = \langle V_T, [2], att_T, lab_T, ext_T \rangle$，满足 $rk_T(1) = rk_T(2) = rk(T) = k$。换言之，T 有两条秩相同的超边，且该秩与 T 的秩一致。下文中，$T(H_1, H_2)$ 是 $T\ [1/H1, 2/H_2]$ 的简写记法。

定义 4. 秩为 k 的模板 T 是幺半群的，如果对于所有秩为 k 的超图 A、B、C，以下条件均成立：1. T (A, T (B,C)) ≅T (T (A,B),C)，2. 存在一个秩为 k 的超图 U_T，使得 T (U_T, A) ≅T (A,U_T) ≅A。
定义 5。类型 A 的 T-迭代 T^n(A)（其中 T 是一个幺半群模板）定义如下，其中 rk(A) =rk(T)：T^0(A) :=U_T；T^{n+1}(A) :=T (T^n(A), A•)（对于 n ≥0）。
例 8。幺半群模板的两个例子是

- $O = \boxed{X}\ \boxed{X}$ (i.e., $V_O = \emptyset$, $E_O = [2]$, $att_O(1) = att_O(2) = ext_O = \Lambda$). Note that $U_O = D[0]$.
- $Str = {}_{(1)}\bullet \xrightarrow{Y} \bullet \xrightarrow{Y} \bullet_{(2)}$.

这里 X、Y 是任意标签，它们并不重要。注意，对于零秩 H、G，O(H, G) =H +G。因此，O^m(A) =m ·A•（其中 rk(A) =0）。

利用幺半群模板，我们可以定义超图合取 Kleene 星。带有合取 Kleene 星 *HLω的超图 Lambek演算的类型按照第 2.2 节中的描述构造，但我们在定义中增加一项：如果 A 是满足 rk(A) =n 的类型，且 T 是一个秩为 n 的幺半群模板，那么 $^*_T$A 也是一个秩为 n 的类型。我们还为这一新运算增加两条推理规则：

$$\frac{(H \to \times(T^n(A)))_{n=0}^{\infty}}{H \to {}^*_T A}\ (\to^*)_\omega \qquad \frac{G[e/T^n(A)] \to B}{G[e/({}^*_T A)^\bullet] \to B}\ ({}^*\to),\ n \geq 0$$

关于 *HLω，会出现一些常见的逻辑问题。特别是，割消去定理可以证明如下：
定理 4. 若 *HL$_\omega \vdash$ H →A 且 *HL$_\omega \vdash$ G[e/A•] →B，则 *HL$_\omega \vdash$ G[e/H] →B .

该定理的证明采用超限归纳法，与 [12] 中的方法类似。

请注意，我们可以用与 $^*_T$A 相同的方式，定义超图克莱尼星，将 * 中研究的 A* 推广开来。尽管合取 Kleene 星有些奇怪且无用，但我们认为，模板、T-迭代等的定义是有用的，因为借助它们，我们可以定义并研究超图克莱尼星。

回到 *HLω，现在我们可以定义 *HLω-文法，并重复构造 3：
构造 5. 设 HGr =⟨N Σ,P,S•⟩ 是一个规范化文法。那么 LGω(HGr) =⟨Σ, S′,▷⟩，其中▷由对 a ▷ T$_a$ 的元素组成，而 $S' = S \div \left(\sum_{r \in P_N} {}^*(o\mathrm{DPO}(r))^\bullet + \$_0^\bullet \right)$.

在这里，我们对每个类型 DPO(r)（其中 r ∈P$_N$）应用超图合取克莱尼星，并将结果存储在 S′ 中。这一技巧使我们能够证明以下结论
定理 5：如果 HGr 是规范化 DPO 文法，那么 L(LGω(HGr)) = L(HGr)。

6 结论

在这项工作中，我们在建立 DPO 文法与 HL 文法之间的联系方面迈出了几步。DPO 文法代表图文法领域中的基于规则的方法，而 HL 文法则是类型逻辑方法的代表。我们证明，任何 DPO 文法都可以通过对 [3] 中方法的推广，变换为等价的 HMEL$_0$-文法。随后，我们对 DPO 文法中的推导施加限制，并证明满足这一限制的文法可以转换为等价的 HL 文法。我们认为，这一限制很有前景：我们主张线性受限 DPO 文法等价于 HL 文法。最近，我们成功地运用同样的思路，对推导施加线性限制，解决了一个关于带置换的Lambek演算 [14] 的开放问题。未来，我们将确立 HL 文法与 DPO 文法之间的精确联系。

致谢

我感谢 Mati Pentus 教授和 Stepan L. Kuznetsov 的富有成效的讨论，也感谢匿名审稿人在各轮评审中提出的宝贵意见（尤其是建议研究与过程代数的联系，这促使我们开始研究另一个课题）。

参考文献

[1] Frank Drewes、Hans-J¨org Kreowski 和 Annegret Habel（1997）：超边替换图文法。载于 Grzegorz Rozenberg（编）：《图文法与图变换计算手册》，第 1 卷：基础，世界科技出版社，第 95–162 页，doi:10.1142/9789812384720 0002。[2] Hartmut Ehrig、Michael Pfender 和 Hans J¨ urgen Schneider（1973）：图文法：一种代数方法。载于：第十四届开关与自动机理论年度研讨会，美国艾奥瓦州爱荷华城，1973 年 10 月 15—17 日，IEEE计算机学会，第 167–180 页，doi:10.1109/SWAT.1973.11。[3] Makoto Kanazawa（1992）：加入附加联结词的 Lambek 演算。《逻辑、语言与信息杂志》1(2)，第 141–171 页，doi:10.1007/BF00171695。[4] Makoto Kanazawa（1999）：Lambek 演算：识别能力与复杂度。载于 Jelle Gerbrandy、Maarten Marx、Maarten de Rijke 和 Yde Venema（编）：《纪念 Johan van Benthem 五十寿辰论文集》，阿姆斯特丹大学出版社。[5] Max I. Kanovich、Stepan L. Kuznetsov、Vivek Nigam 和 Andre Scedrov（2020）：软次指数与复用。载于：IJCAR（1），《计算机科学讲义》12166，施普林格，第 500–517 页，doi:10.1007/978-3-030-51074-9 29。[6] Barbara K¨onig、Dennis Nolte、Julia Padberg 和 Arend Rensink（2018）：图变换教程。载于 Reiko Heckel 和 Gabriele Taentzer（编）：《图变换、规约与网——纪念 Hartmut Ehrig》，《计算机科学讲义》10800，施普林格，第 83–104 页，doi:10.1007/978-3-319-75396-6 5。[7] Stepan L. Kuznetsov（2021）：带 Kleene 星号的无穷 Lambek 演算的复杂度。《符号逻辑评论》14(4)，第 946–972 页，doi:10.1017/S1755020320000209。[8] Joachim Lambek（1958）：句子结构的数学。《美国数学月刊》65(3)，第 154–170 页，doi:10.1080/00029890.1958.11989160。[9]Patrick Lincoln、John C. Mitchell、Andre Scedrov 和 Natarajan Shankar（1992）：命题线性逻辑的判定问题。《纯粹与应用逻辑年鉴》56(1-3)，第 239–311 页，doi:10.1016/0168-0072(92)90075-B。[10] Franco Montagna（2004）：多值逻辑中的存储算子与乘法量词。《逻辑计算杂志》14(2)，第 299–322 页，doi:10.1093/logcom/14.2.299。[11] Richard Moot 和 Mario Piazza（2001）：一阶直觉主义线性逻辑的语言学应用。《逻辑、语言与信息杂志》10(2)，第 211–232 页，doi:10.1023/A:1008399708659。[12] Ewa Palka（2007）：*-连续作用格等式理论的无穷序列演算系统。《基础信息学》78(2)，第 295–309 页。[13] Tikhon Pshenitsyn（2021）：基于超图语言逻辑的文法。载于 Berthold Hoffmann 和 Mark Minas（编）：第十二届国际图计算模型研讨会论文集，GCM@STAF2021，线上，2021 年 6 月 22 日，EPTCS350，第 1–18 页，doi:10.4204/EPTCS.350.1。[14] Tikhon Pshenitsyn（2022）：带置换的 Lambek 演算上的文法：识别能力及其与带状态分支向量加法系统的联系，doi:10.48550/ARXIV.2209.08415。可见于 https://arxiv.org/abs/2209.08415。[15] Tikhon Pshenitsyn（2022）：超图兰贝克文法。《逻辑与代数编程方法杂志》129，文章编号 100798，doi:10.1016/j.jlamp.2022.100798。