Transformation of DPO Grammars into Hypergraph Lambek Grammars With The Conjunctive Kleene Star

Tikhon Pshenitsyn
Department of Mathematical Logic and Theory of Algorithms, Faculty of Mathematics and Mechanics *
Lomonosov Moscow State University, GSP-1, Leninskie Gory, Moscow, 119991, Russian Federation
ptihon at yandex.ru

We study how to embed well-known hypergraph grammars based on the double pushout (DPO) approach in the hypergraph Lambek calculus HL. It turns out that DPO rules can be naturally encoded by types of HL. However, this encoding is not enough to convert a DPO grammar into an equivalent grammar based on HL: we additionally need a logical operation that would allow making arbitrarily many copies of types. We develop such an operation called the conjunctive Kleene star and show that any DPO grammar can be converted into an equivalent HL-grammar enriched with this operation.

1 Introduction

In this paper, we aim to compare two kinds of graph grammars: DPO hypergraph grammars and hypergraph Lambek grammars. More precisely, we focus on embedding the former formalism in the latter.

DPO hypergraph grammars are one of the most well-known kinds of graph grammars; they were introduced in 1973 [2]. They are designed to generalize Chomsky formal grammars from strings to graphs. Recall that a production in a formal grammar of the form $\alpha \Rightarrow \beta$ allows one to replace a substring α in any string γ by a string β. A production of a DPO hypergraph grammar, in turn, can be presented in the form $L \Rightarrow R$ where L and R are two hypergraphs. The procedure of replacing a hypergraph by another hypergraph, however, needs further clarification; this is done by using the double pushout approach, which is widely used in the field of graph grammars.

The hypergraph Lambek calculus HL and hypergraph Lambek grammars are novel approaches described in [8]. They are based on logical grounds: HL generalizes the Lambek calculus introduced in [6]. The Lambek calculus L is a substructural logic of intuitionistic logic, and it is designed to model syntax of natural languages. The hypergraph Lambek calculus HL inherits main principles of L, its structural and model-theoretic properties. Besides, HL forms the basis for hypergraph Lambek grammars (HL-grammars). An HL-grammar is defined by an assignment of a finite number of types of HL to symbols of an alphabet; in order to check that a terminal hypergraph H is generated by a grammar one needs to replace each symbol in H by one of the types corresponding to it (which results in a hypergraph labeled by types) and to check that the resulting structure is derivable from axioms by rules of HL.

Our objective is to figure out what class of hypergraph languages HL-grammars generate and how they are related to other kinds of graph grammars. In particular, it is clearly important to compare them with widely studied DPO grammars. The following question arises: can we convert each DPO grammar into an HL-grammar generating the same language? A simple complexity reason shows that the answer is negative: DPO grammars are universal while the membership problem for HL-grammars

*The study was funded by RFBR, project N20-01-00670; the Interdisciplinary Scientific and Educational School of Moscow University "Brain, Cognitive Systems, Artificial Intelligence"; the Theoretical Physics and Mathematics Advancement Foundation "BASIS".

Submitted to:
TERMGRAPH 2022

is NP-complete. Nevertheless, it turns out that there is a way to naturally encode any DPO rule as a type of HL; however, in order to apply a rule arbitrarily many times using this encoding one needs a logical operation in HL that would allow copying types in HL arbitrarily many times. Note that, in the string case, an extension of the Lambek calculus with Kleene star is studied in [5, 7]. Kleene star provides a possibility to copy a type arbitrarily many times, but the way of copying does not suit our specific needs when considering DPO rules within HL. This leads us to defining a related operation, which we call *the conjunctive Kleene star* and denote as follows: ${}^*A$. The hypergraph Lambek calculus with the conjunctive Kleene star ${}^*\mathrm{HL}_\omega$ defined in Section 3 perfectly suits our needs: any DPO grammar can be transformed into an equivalent ${}^*\mathrm{HL}_\omega$-grammar.

2 Preliminaries

Σ^* is the set of strings over an alphabet Σ including the empty word Λ; if R is a relation, then R^* is its transitive reflexive closure. Each function $f : \Sigma \to \Delta$ can be extended to a homomorphism $f : \Sigma^* \to \Delta^*$. By $w(i)$ we denote the i-th symbol of w. Let $[n]$ denote the set $\{1,2,\dots,n\}$ (and $[0] := \emptyset$).

Given a set of *labels* Σ along with a *rank function* $rk : \Sigma \to \mathbb{N}$, a *hypergraph* G over Σ is a tuple $G = \langle V_G, E_G, att_G, lab_G, ext_G\rangle$ where V_G is a finite set of *nodes*, E_G is a finite set of *hyperedges* (V_G and E_G are always considered to be disjoint), $att_G : E_G \to V_G^*$ assigns a string (understand it as an ordered multiset) of *attachment nodes* to each hyperedge, $lab_G : E_G \to \Sigma$ labels each hyperedge by some element of Σ in such a way that $rk(lab_G(e)) = |att_G(e)|$ whenever $e \in E_G$, and $ext_G \in V_G^*$ is a string of *external nodes*. Hypergraphs are always considered up to isomorphism. The set of all hypergraphs with labels from Σ is denoted by $\mathscr{H}(\Sigma)$. Note that we allow attachment nodes of a hyperedge as well as external nodes to coincide. The *rank function* rk_G (or rk, if G is clear) is defined as follows: $rk_G(e) := |att_G(e)|$. Besides, $rk(G) := |ext_G|$.

In drawings of hypergraphs, black circles correspond to nodes, labeled rectangles correspond to hyperedges, *att* is represented by numbered lines, and external nodes are represented by numbers in brackets. If a hyperedge has exactly two attachment nodes, it is depicted by a labeled arrow that goes from the first attachment node to the second one.

A *handle* $a^\bullet$ is a hypergraph $a^\bullet = \langle [n], [1], att, lab, 1\dots n\rangle$ where $att(1) = 1\dots n$ and $lab(1) = a$ ($a \in \Sigma$, $rk(a) = n$). A hypergraph a° is of the form $\langle [n], [1], att, lab, \Lambda\rangle$ where *att*, *lab* are as in the definition of $a^\bullet$. A hypergraph $\mathrm{D}[k] = \langle [k], \emptyset, \emptyset, \emptyset, \Lambda\rangle$ is called *discrete* ($k \in \mathbb{N}$).

For a hypergraph H and a function $f : E_H \to \Sigma$ *a relabeling* $f(H)$ *of* H is a hypergraph $f(H) = \langle V_H, E_H, att_H, f, ext_H\rangle$. It is required that $rk_H(e) = rk(f(e))$ for $e \in E_H$.

The *replacement* of a hyperedge e_0 in G by a hypergraph H (such that $rk(e_0) = rk(H)$) is done as follows: 1. remove e_0 from G; 2. insert an isomorphic copy of H (H and G have to consist of disjoint sets of nodes and hyperedges); 3. for each i, fuse the i-th external node of H with the i-th attachment node of e_0. The result is denoted as $G[e_0/H]$. It is well known that if several hyperedges of a hypergraph are replaced by other hypergraphs, then the result does not depend on the order of the replacements; moreover the result is not changed, if replacements are done simultaneously [1]. The following notation is in use: if $e_1,\dots,e_k$ are distinct hyperedges of a hypergraph H and they are simultaneously replaced by hypergraphs $H_1,\dots,H_k$ resp., then the result is denoted $H[e_1/H_1,\dots,e_k/H_k]$.

In a special case where a hypergraph G does not have external nodes ($ext_G = \Lambda$) let us call it *zero-rank* and denote it using four components: $G = \langle V_G, E_G, att_G, lab_G\rangle$. The following definitions are applicable for zero-rank hypergraphs H, H_1, H_2. The *disjoint union* $H_1 + H_2$ is the hypergraph $\langle V_{H_1} \sqcup V_{H_2}, E_{H_1} \sqcup E_{H_2}, att, lab\rangle$ such that $att|_{H_i} = att_{H_i}$, $lab|_{H_i} = lab_{H_i}$ ($i = 1,2$); that is, we just put these hypergraphs

together without fusing any nodes or hyperedges. The disjoint union of H with itself k times is denoted by $k \cdot H$. Besides, we can extend the notion of disjoint union: if H_1 is zero-rank, then we can define $H_1 + H_2$ and $H_2 + H_1$ assuming that the disjoint union has the set of external nodes equal to ext_{H_1}.

2.1 DPO Grammars

Given two zero-rank hypergraphs G and H, a *morphism* $f : G \to H$ is a pair of functions $f_V : V_G \to V_H$, $f_E : E_G \to E_H$ such that $f_V(att_G(e)) = att_H(f_E(e))$, $lab_H(f_E(e)) = lab_G(e)$ for all $e \in E_G$.

Let I, G_1, G_2 be zero-rank hypergraphs with morphisms $\varphi_i : I \to G_i$, $i = 1,2$. Let $\equiv_V$ be the smallest equivalence relation on the disjoint union $V_{G_1} \sqcup V_{G_2}$ that satisfies $\varphi_1(v) \equiv \varphi_2(v)$ for $v \in V_I$; a relation $\equiv_E$ is defined similarly on $E_{G_1} \sqcup E_{G_2}$. $\langle x \rangle$ denotes the equivalence class of x w.r.t. $\equiv_V$ if x is a node, and w.r.t. $\equiv_E$ if x is a hyperedge. The *gluing* of G_1 and G_2 over I denoted as $G_1 +_{\varphi_1,\varphi_2} G_2$ is a hypergraph G such that $V_G = (V_{G_1} \sqcup V_{G_2})/\equiv_V$, $E_G = (E_{G_1} \sqcup E_{G_2})/\equiv_E$; given $\langle e \rangle \in E_G$ with $rk(e) = k$, if $e \in E_{G_i}$ for some $i = 1,2$, then $att_G(\langle e \rangle) = \langle att_{G_i}(e)(1) \rangle \dots \langle att_{G_i}(e)(k) \rangle$ and $lab_G(\langle e \rangle) = lab_{G_i}(e)$. This is a well-defined notion taken from [4] (where it is defined for graphs rather than for hypergraphs). There it is stated that the gluing of two graphs is a pushout in the category of graphs. In this paper, we do not work within the categorical approach, so we stick to the set-theoretic definition.

Note that, if I is discrete, then the gluing procedure can be represented as replacement:

Proposition 1. *Let $I = \mathrm{D}[k]$ and let G_i, φ_i be as above. Let $G_1' = \langle V_{G_1}, E_{G_1} \sqcup \{e_0\}, att_{G_1'}, lab_{G_1'} \rangle$ where $att_{G_1'}(e) = att_{G_1}(e)$, $lab_{G_1'}(e) = lab_{G_1}(e)$ for $e \in E_{G_1}$, and $att_{G_1'}(e_0) = \varphi_1(1) \dots \varphi_1(k)$ (the label of e_0 does not matter). Let $G_2' = \langle V_{G_2}, E_{G_2}, att_{G_2}, lab_{G_2}, \varphi_2(1) \dots \varphi_2(k) \rangle$. Then $G_1 +_{\varphi_1,\varphi_2} G_2 = G_1'[e_0/G_2']$.*

This proposition immediately follows from the definitions of gluing and replacement.

In the DPO approach, a *hypergraph grammar rule* is of the form $r = (L \xleftarrow{\varphi_L} I \xrightarrow{\varphi_R} R)$ where φ_L, φ_R are morphisms. A hypergraph G *is transformed into* H *via* r if there is a hypergraph C and a morphism $\psi : I \to C$ such that $G \cong C +_{\psi,\varphi_L} L$, $H \cong C +_{\psi,\varphi_R} R$ [4] ($\cong$ means that hypergraphs are isomorphic). We draw the reader's attention to the fact that this can be expressed by a double pushout diagram:

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

This transformation is denoted as $G \underset{r}{\Rightarrow} H$ or simply as $G \Rightarrow H$. NB! Hereinafter, we consider only hypergraph rules with I being discrete. This does not substantially restrict the formalism.

Example 1. Consider the following DPO rule ρ:

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

Note that both the leftmost and the rightmost hypergraphs are zero-rank; numbers in braces represent images of nodes of the interface hypergraph $\mathrm{D}[3]$ (i.e. $\varphi_L(1)$ is the node with the superscript $\{1\}$ in the leftmost hypergraph and so on).

A *DPO hypergraph grammar* HGr is of the form $\langle N, \Sigma, P, Z \rangle$ where N, Σ are disjoint finite alphabets of *nonterminal* and *terminal* labels resp., P is a finite set of hypergraph rules over $N \cup \Sigma$, and Z is a zero-rank *start hypergraph*. The language $L(HGr)$ generated by HGr is the set of all zero-rank hypergraphs $H \in \mathscr{H}(\Sigma)$ such that $Z \Rightarrow^* H$. Note that we can assume without loss of generality that $Z = S^\circ$ for $S \in N$.

2.2 Hypergraph Lambek Calculus and Hypergraph Lambek Grammars

Now let us define the hypergraph Lambek calculus. We fix a set Pr of *primitive types* along with a function $rk : Pr \to \mathbb{N}$; we require that for each $k \in \mathbb{N}$ there are infinitely many $p \in Pr$ such that $rk(p) = k$. Besides, we fix a countable set of labels $\$_n, n \in \mathbb{N}$ and set $rk(\$_n) = n$; let us agree that these labels do not belong to any other set considered in the definition of the calculus. Then the set of *types* Tp is defined inductively as follows:

1. All primitive types are types.
2. Let $N \in Tp$ be a type, and let D be a hypergraph such that labels of all its hyperedges, except for one, are from Tp, and one of them equals $\$_d$ for some d; let also $rk(N) = rk(D)$. Then $N \div D$ is also a type such that $rk(N \div D) := d$. The hyperedge of D labeled by $\$_d$ is denoted by $e_D^{\$}$.
3. If $M \in \mathscr{H}(Tp)$ is a hypergraph labeled by types, then $\times(M)$ is also a type, and $rk(\times(M)) := rk(M)$.

Example 2 contains an exemplar of a type. A *sequent* is a structure of the form $H \to A$ where H is a hypergraph labeled by types (called the *antecedent* of the sequent), and A is a type (called the *succedent*) such that $rk(H) = rk(A)$.

The hypergraph Lambek calculus HL deals with hypergraph sequents. The only axiom of HL is of the form $p^{\bullet} \to p$ where $p \in Pr$. There are four inference rules of HL:

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

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

Here $N \div D$, $\times(M)$ are types; $e \in E_H$; $E_D = \{e_D^{\$}, d_1, \dots, d_k\}$, $E_M = \{m_1, \dots, m_l\}$. In each rule presented above, the sequents above the line are called *premises*, and the sequent below the line is called the *conclusion*. A hypergraph sequent $H \to A$ is said to be *derivable in* HL (denoted by $\mathrm{HL} \vdash H \to A$) if it can be obtained from axioms of HL by applications of rules of HL. A corresponding sequence of rule applications is called a *derivation*. An example of a derivation is given in Example 3. Motivation of the introduced rules is explained in [8].

An HL-*grammar* is a tuple $HGr = \langle \Sigma, S, \triangleright \rangle$ where Σ is an alphabet along with a rank function, $S \in Tp$ is a distinguished type, and $\triangleright \subseteq \Sigma \times Tp$ is a finite binary relation such that $a \triangleright T$ implies $rk(a) = rk(T)$. *The language* $L(HGr)$ *generated by an* HL-*grammar* $HGr = \langle \Sigma, S, \triangleright \rangle$ is the set of all hypergraphs $G \in \mathscr{H}(\Sigma)$ for which a function $f_G : E_G \to Tp$ exists such that:

1. $lab_G(e) \triangleright f_G(e)$ whenever $e \in E_G$;
2. $\mathrm{HL} \vdash f_G(G) \to S$ (recall that $f_G(G)$ is a relabeling of G by means of f_G).

3 Encoding DPO Rules in HL With Conjunctive Kleene Star

Inference rules of the hypergraph Lambek calculus are defined as transformations operating on hypergraph sequents. All the rules are defined through replacement; besides, after an application of each rule a new type appears either in the antecedent or in the succedent of a sequent. Let us take a closer look at two particular rules, namely, at $(\div \to)$ and $(\times \to)$. The rule $(\div \to)$ is organized as follows: given a sequent $H[e/N^{\bullet}] \to A$ (note that $H[e/N^{\bullet}]$ is structurally the same hypergraph H, the replacement only

changes the label of e) and sequents $H_i \to lab_D(d_i)$ for $i = 1,\dots,k$, we replace e in H by D, then relabel the $\$_d$-labeled hyperedge by the type $(N \div D)$, and then replace each d_i by the corresponding antecedent H_i ($i = 1,\dots,k$). Hence, this rule essentially consists of several replacements. In contrast, the rule $(\times \to)$ performs a transformation inverse to replacement: if one has a hypergraph $G[e/M]$ in the antecedent, then he/she can "compress" its subhypergraph M into a single hyperedge e labeled by the type $\times(M)$.

Note that the definition of a hypergraph grammar rule and Proposition 1 imply that the rule application of $G \underset{r}{\Rightarrow} H$ for $r = (L \overset{\varphi_L}{\leftarrow} I \overset{\varphi_R}{\to} R)$ consists of an inverse replacement $G = C'[e_0/L'] \Leftarrow C'$ and of a replacement $C' \Rightarrow C'[e_0/R'] = H$ (where C is as in the definition of a hypergraph rule application, and C', e_0, L' and R' are as in Proposition 1). This shows us a way of encoding each DPO hypergraph rule by a type of HL with $\times$ and $\div$. However, firstly we need to slightly enhance DPO hypergraph grammars.

Construction 1. Given a DPO hypergraph grammar $\langle N, \Sigma, P, S^\circ \rangle$, we convert it into an equivalent grammar $\langle N', \Sigma, P', S^\circ \rangle$, which we call *normalized*, as follows. For each $a \in \Sigma$ we introduce a new nonterminal label T_a with $rk(T_a) = rk(a)$; let $N' = N \sqcup \{T_a \mid a \in \Sigma\}$. Then for each $r = (L \overset{\varphi_L}{\leftarrow} \mathrm{D}[k] \overset{\varphi_R}{\to} R) \in P$ we replace each terminal label a in L, R by T_a. Let us call such new rules *nonterminal* and denote the set of nonterminal rules as P_N. Finally, we add rules that allow one to replace T_a by a, i.e., rules of the form $(T_a^\circ \overset{\varphi_L}{\leftarrow} \mathrm{D}[k] \overset{\varphi_R}{\to} a^\circ)$ where $rk(a) = k$, $\varphi_L(i) = \varphi_R(i) = i$ for $i = 1,\dots,k$ (here we use the notation of nodes as in the definitions of $\mathrm{D}[k]$ and S°). These rules are called *terminal* and are denoted as P_T. Finally, $P' = P_N \cup P_T$. Hereinafter we consider only normalized grammars.

Now we are ready to convert a hypergraph grammar rule into a corresponding type of HL.

Construction 2. Let us consider nonterminal labels of normalized grammars as primitive types (with the same rank function). If $r = (L \overset{\varphi_L}{\leftarrow} \mathrm{D}[k] \overset{\varphi_R}{\to} R)$ is a nonterminal rule, then $\mathrm{DPO}(r) := \times(\widehat{L}) \div \widehat{R}$ where

1. $\widehat{L} = \langle V_L, E_L, att_L, lab_L, \varphi_L(1) \dots \varphi_L(k) \rangle$;
2. $\widehat{R} = \langle V_R, E_R, att_R, lab_R, \varphi_R(1) \dots \varphi_R(k) \rangle + \$_0^\bullet$.

Note that $\$_0^\bullet$ is a separate hyperedge of rank 0 "floating" in $\widehat{R}$.

Example 2. The nonterminal rule ρ from Example 1 is converted into the following type $\mathrm{DPO}(\rho)$:

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

The main connection between r and $\mathrm{DPO}(r)$ is shown in

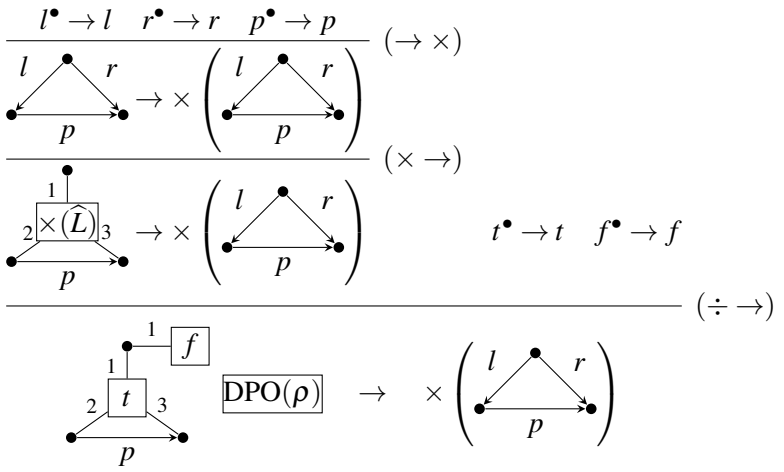
Lemma 1. *Let Y be a zero-rank hypergraph and let $Y \underset{r}{\Rightarrow} Y'$ for $r \in P_N$. If $\mathrm{HL} \vdash Y \to A$ for some type A, then $\mathrm{HL} \vdash Y' + \mathrm{DPO}(r)^\bullet \to A$ as well.*

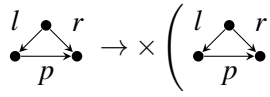
Lemma 2. *Let Y, Y' be zero-rank; let $Y \Rightarrow^* Y'$ in a normalized grammar $\langle N, \Sigma, P, S^\circ \rangle$ by means of nonterminal rules. If $\mathrm{HL} \vdash Y \to A$ for some type A, then $\mathrm{HL} \vdash Y' + \sum\limits_{r \in P_N} k_r \cdot \mathrm{DPO}(r)^\bullet \to A$ for some $k_r \in \mathbb{N}$.*

Here the summation symbol stands for multiple disjoint union. Lemma 1 is proved by straightforwardly applying $(\times \to)$ and then $(\div \to)$ to the sequent $Y \to A$. Lemma 2 directly follows from Lemma 1. Note that k_r is the number of applications of r in the derivation $Y \Rightarrow^* Y'$.

Therefore, a DPO derivation can be remodeled within HL but each rule application of r leaves a trace, namely, a floating hyperedge labeled by $\mathrm{DPO}(r)$ in the antecedent.

Example 3. The following derivation illustrates Lemma 1:



Here the sequent $Y \to A$ equals  , and the rule r equals ρ from Example 1.

The next goal is to use the correspondence between rules and types at the grammar level:

Definition 1. Given a DPO hypergraph grammar $HGr = \langle N, \Sigma, P, S^\circ \rangle$, let $L_c(HGr)$ consist of all hypergraphs $H \in L(HGr)$ such that there exists a derivation $S^\circ \Rightarrow^* H$ with no more than $c \cdot |E_H|$ steps.

Construction 3. Let $HGr = \langle N, \Sigma, P, S^\circ \rangle$ be a normalized grammar; let $c \in \mathbb{N}$. Then we construct an HL-grammar $\mathrm{LG}_c(HGr) = \langle \Sigma, \times(S^\circ), \triangleright \rangle$ where $\triangleright$ contains exactly the following pairs:

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

Note that $\triangleright$ is a finite relation since there are finitely many $k_r \in \mathbb{N}$ satisfying the above requirements.

Example 4. Consider a DPO grammar $HGr = \langle N, \Sigma, P, S^\circ \rangle$ where $N = \{S\}$, $\Sigma = \{a\}$, and $P = \{r_1, r_2, r_3\}$:

1. $r_1 = \Big(\boxed{S} \leftarrow \mathrm{D}[0] \rightarrow \mathrm{D}[0] \Big)$;
2. $r_2 = \Big(\boxed{S} \leftarrow \mathrm{D}[0] \rightarrow \boxed{S} \ \bullet \Big)$;
3. $r_3 = \Big({}_{\{1\}}\bullet \quad \bullet_{\{2\}} \leftarrow \mathrm{D}[2] \rightarrow {}_{\{1\}}\bullet \xrightarrow{a} \bullet_{\{2\}} \Big)$.

It is not hard to see that it generates all graphs (with edges having two attachment nodes): the rule r_2 produces nodes while r_3 produces edges. Consider e.g. the following derivation:

$$\boxed{S} \underset{r_2}{\Rightarrow} \boxed{S} \bullet \underset{r_2}{\Rightarrow} \boxed{S} \bullet \bullet \underset{r_1}{\Rightarrow} \bullet \quad \bullet \underset{r_3}{\Rightarrow} \bullet \xrightarrow{a} \bullet \underset{r_3}{\Rightarrow} \bullet \xrightarrow{a} \bullet \circlearrowleft a \underset{r_3}{\Rightarrow} a \circlearrowright \bullet \xrightarrow{a} \bullet \circlearrowleft a \tag{1}$$

Note that HGr is not normalized; using Construction 1 we replace r_3 by the following two rules:

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

Let us denote a new normalized grammar HGr'. Then we convert its nonterminal rules into types by using Construction 2:

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

Finally, we introduce an HL-grammar $\mathrm{LG}_2(HGr') = \langle \Sigma, \times(S^\circ), \triangleright \rangle$ according to Construction 3. The binary relation $\triangleright$ consists of the following 13 pairs (in fact, of the 10 distinct pairs) where $i, j \in \{1,2,3\}$:

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

Recall that in order to check that a hypergraph belongs to $L(\mathrm{LG}_2(HGr'))$ we need 1) to replace labels of its hyperedges by types corresponding to them via $\triangleright$; 2) to construct a sequent with the resulting hypergraph in the antecedent and with $\times(S^\circ)$ in the succedent; 3) to derive this sequent. Let us check that $H = \; a \circlearrowright\bullet\xrightarrow{a}\bullet\circlearrowleft a \; \in L(\mathrm{LG}_2(HGr'))$. We replace each label a by one of the types T, T_i, or T_{ij} as follows: $T_{22} \circlearrowright\bullet\xrightarrow{T_{13}}\bullet\circlearrowleft T_{33}$ (compare the indices of types with the numbers of rules applied in (1)). Finally, we check that $T_{22} \circlearrowright\bullet\xrightarrow{T_{13}}\bullet\circlearrowleft T_{33} \to \times(S^\circ)$ is derivable:

$$\dfrac{\dfrac{\dfrac{\dfrac{T_a \circlearrowright\bullet\xrightarrow{T_a}\bullet\circlearrowleft T_a \quad \boxed{X_2}\,\boxed{X_2}\,\boxed{X_1}\,\boxed{X_3}\,\boxed{X_3}\,\boxed{X_3} \quad \to \quad \times(S^\circ)}{T_a \circlearrowright\bullet\xrightarrow{T_a}\bullet\circlearrowleft T_{33} \quad \boxed{X_2}\,\boxed{X_2}\,\boxed{X_1}\,\boxed{X_3} \quad \to \quad \times(S^\circ)}(\times\to)}{T_a \circlearrowright\bullet\xrightarrow{T_{13}}\bullet\circlearrowleft T_{33} \quad \boxed{X_2}\,\boxed{X_2} \quad \to \quad \times(S^\circ)}(\times\to)}{T_{22} \circlearrowright\bullet\xrightarrow{T_{13}}\bullet\circlearrowleft T_{33} \quad \to \quad \times(S^\circ)}(\times\to)$$

The uppermost sequent in the above derivation is derivable, which follows from Lemma 2. This completes the proof of the fact that $H \in L(\mathrm{LG}_2(HGr'))$.

The above example illustrates the following theorem:

Theorem 1. *If HGr is a normalized DPO grammar and $1 \le c \in \mathbb{N}$, then $L(\mathrm{LG}_{c-1}(HGr)) = L_c(HGr)$.*

The inclusion $L(\mathrm{LG}_{c-1}(HGr)) \supseteq L_c(HGr)$ is proved by using Lemma 2 in the same way as in Example 4. The other inclusion is proved by induction; namely, the following proposition is crucial:

Proposition 2. *If $\mathrm{HL} \vdash H + \sum\limits_{r \in P_N} k_r \cdot \mathrm{DPO}(r)^\bullet \to B$ where H is labeled by nonterminal symbols, $k_r \in \mathbb{N}$, and B is a nonterminal symbol, then for each zero-rank hypergraph G with $e_0 \in E_G$ such that $rk(e_0) = rk(B)$ it holds that $G[e_0/B^\bullet] \Rightarrow^* G[e_0/H]$.*

Theorem 1 says that HL-grammars are powerful enough to generate hypergraphs of a language generated by a DPO grammar such that the number of steps in their derivation is bounded by a linear function of the number of hyperedges. It might be the case for a DPO grammar HGr that $L(HGr) = L_c(HGr)$ for some $c \in \mathbb{N}$; in fact, we claim that for each HL-grammar $HGr = \langle \Sigma, S, \triangleright \rangle$ with $rk(S) = 0$ there is a DPO grammar HGr' and $c \in \mathbb{N}$ such that $L(HGr) = L_c(HGr') = L(HGr')$, although we do not prove this (this should be a matter of another paper). In general, however, $L(HGr) \neq L_c(HGr)$ (e.g. in Example 4 $L_{k+1}(HGr)$ contains only graphs G such that $|V_G| < k \cdot |E_G|$). Besides, it follows from complexity reasons that an arbitrary DPO grammar cannot be converted into an equivalent HL-grammar: it is known that the membership problem even for a particular DPO grammar is undecidable, while it is NP-complete for HL-grammars. However, Construction 2 is quite natural, so we would like to modify the hypergraph Lambek calculus somehow in order to be able to convert any DPO grammar into an equivalent HL-grammar.

The idea of Construction 3 and of Theorem 1 is that we store $\mathrm{DPO}(r)$-labeled hyperedges in each type corresponding to a terminal symbol; then, for $G \in L(\mathrm{LG}_c(HGr))$, when we replace each symbol a in G by a type $\times \left(T_a^\bullet + \sum\limits_{r \in P_N} k_r \cdot \mathrm{DPO}(r)^\bullet \right)$ for some k_r, these hyperedges eventually appear in the antecedent 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 $L(HGr)$. To overcome this limitation we need a way of copying types in antecedents of sequents unlimitedly. Since HL generalizes the Lambek calculus, we expect that this way must have logical and algebraic grounds.

Any logical calculus is defined syntactically: we must introduce notions of logical formulae, which are built using some operations, then axioms and inference rules of a calculus (recall e.g. the classical propositional calculus). The Lambek calculus L introduced in [6] follows the same scheme. There are different modifications of L; let us start with considering *the Lambek calculus with the unit* denoted as $\mathrm{L}_{\mathbf{1}}$, which is the logic of residuated monoids. Types of $\mathrm{L}_{\mathbf{1}}$ are built from primitive types Pr (here we do not need the rank function) using binary operations $\backslash$, $\cdot$, $/$; there is also a constant $\mathbf{1}$ called the unit. A sequent is a structure of the form $A_1, \ldots, A_n \to A$ where $n \geq 0$ and A_i, A are types. Axioms of the calculus $\mathrm{L}_{\mathbf{1}}$ are $A \to A$ and $\to \mathbf{1}$. There are seven rules of $\mathrm{L}_{\mathbf{1}}$:

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

$$\frac{\Gamma, A, B, \Delta \to C}{\Gamma, A \cdot B, \Delta \to C}\ (\cdot \to) \qquad \frac{\Pi \to A \quad \Psi \to B}{\Pi, \Psi \to A \cdot B}\ (\to \cdot) \qquad \frac{\Gamma, \Delta \to C}{\Gamma, \mathbf{1}, \Delta \to C}\ (\mathbf{1} \to)$$

Here capital Latin letters denote types, and capital Greek letters denote sequences of types.

The most common way of modifying L is adding new operations along with new axioms and rules. In particular, one can extend the Lambek calculus with operations corresponding to set-theoretic operations like union or intersection [3]. Another operation, which can be added to L, is Kleene star. It turns out that the Lambek calculus with Kleene star is studied by several researchers, in particular, in [5]. An extension of L by intersection, union, and Kleene star is known as infinitary action logic [7], or, in the algebraic setting, as the logic of action algebras. Note that Kleene star can be described in terms of actions within a transition system: if A is a class of actions, then A^* means actions from A repeated several times [5]. This understanding is very close to what we search for since our goal is to be able to apply DPO rules arbitrarily many times within HL using Construction 2. Hence, let us look at the Lambek calculus with Kleene star $\mathrm{L}^*_{\mathbf{1}\omega}$ [5]. Types of $\mathrm{L}^*_{\mathbf{1}\omega}$ are built from primitive types Pr and from $\mathbf{1}$ using $\backslash, /, \cdot, ^*$. The set of axioms and rules is the same as above but with the following additional rules for Kleene star:

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

Here $A^0 := \mathbf{1}$, $A^{n+1} := A^n \cdot A$. Note that the rule $(\to^*)$ is in fact a countable set of rules for each $n \geq 0$; contrarily, $({}^*\to)_\omega$ is a sinlge rule with countably many premises. Let us clarify the notion of being derivable in this calculus: the set of derivable sequents in $\mathrm{L}^*_{\mathbf{1}\omega}$ is the least set S containing all axioms of $\mathrm{L}^*_{\mathbf{1}\omega}$ (i.e., all sequents of the form $A \to A$ and the sequent $\to \mathbf{1}$) such that it is closed under applications of all inference rules (i.e., if, for some rule, all sequents above the line belong to S, then the sequent below the line must also belong to S). In other words, a derivation in $\mathrm{L}^*_{\mathbf{1}\omega}$ is again a sequence of rule applications, which now can be countable in size but which does not have branches of infinite length.

Unfortunately, the rules for Kleene star work in an undesirable way: they allow unlimited copying types in succedents of sequents (namely, if we have n copies of A in a succedent, then we can wrap them into a single type A^*) but not in antecedents. This motivates us to consider an operation behaving dually:

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

The operation ${}^*A$ is called *the conjunctive Kleene star*. Algebraically, it can be defined in complete residuated lattices using infinitary conjunction as ${}^*a = \bigwedge_{n=0}^{\infty} a^n = \inf\{a^n \mid n \in \mathbb{N}\}$ (this is why we call it conjunctive). Note that the language semantics of this operation is poor: if L is a language ($L \subseteq \Sigma^*$), $\mathbf{1}$ equals $\{\Lambda\}$, and multiplication of languages means pairwise concatenation of their words while conjunction means intersection, then ${}^*L = \{\Lambda\}$ if $\Lambda \in L$ and ${}^*L = \emptyset$ otherwise. Finding a class of residuated lattices, in which the conjunctive Kleene star is defined and is meaningful, is a curious open question.

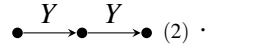
Now we need to generalize the conjunctive Kleene star and inference rules for it to hypergraphs. A question arises: how should one understand an iteration of a type, namely, A^n? In the string case, this means repeating a type n times and writing copies in line connecting them by $\cdot$. We need to extend this iteration procedure to hypergraphs. We suggest the following general definitions:

Definition 2. A *template* T of rank k is a hypergraph $T = \langle V_T, [2], att_T, lab_T, ext_T \rangle$ such that $rk_T(1) = rk_T(2) = rk(T) = k$. In other words, T has two hyperedges of the same rank, which coincides with the rank of T. Hereinafter $T(H_1, H_2)$ is a shorthand notation for $T[1/H_1, 2/H_2]$.

Definition 3. A template T of rank k is *monoidal* if for all hypergraphs A, B, C of rank k it holds that 1. $T(A, T(B,C)) \cong T(T(A,B), C)$, 2. a hypergraph U_T of rank k exists such that $T(U_T, A) \cong T(A, U_T) \cong A$.

Definition 4. The *T-iteration* $T^n(A)$ of a type A (where T is a monoidal template) such that $rk(A) = rk(T)$ is defined as follows: $T^0(A) := U_T$; $T^{n+1}(A) := T(T^n(A), A^\bullet)$ (for $n \geq 0$).

Example 5. Two examples of monoidal templates are

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

Here X, Y are arbitrary labels, they do not matter.

Lemma 3. *$O(H,G) = H + G$ for zero-rank H, G. Consequently, $O^m(A) = m \cdot A^\bullet$ (where $rk(A) = 0$).*

Using monoidal templates we can define the hypergraph conjunctive Kleene star. Types of the hypergraph Lambek calculus with the conjunctive Kleene star ${}^*\mathrm{HL}_\omega$ are built as described in Section 2.2 but we add one more item to the definition: if A is a type such that $rk(A) = n$ and if T is a monoidal template of rank n, then ${}^*_T A$ is also a type of rank n. We also add two inference rules for the new operation:

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

Usual logical questions concerning ${}^*\mathrm{HL}_\omega$ arise. In particular, the cut elimination theorem can be proved:

Theorem 2. *If* ${}^*\mathrm{HL}_\omega \vdash H \to A$ *and* ${}^*\mathrm{HL}_\omega \vdash G[e/A^\bullet] \to B$*, then* ${}^*\mathrm{HL}_\omega \vdash G[e/H] \to B$.

The theorem is proved by a transfinite induction in a similar way to that from [7].

Note that we can define *the hypergraph Kleene star* generalizing A^* studied in [5, 7] in the same way as ${}^*_T A$, which would also be interesting to study as a more classical operation than ${}^*A$.

At the next step we can define ${}^*\mathrm{HL}_\omega$-grammars in the same way as in Section 2.2. Now each DPO grammar can be naturally transformed into an equivalent ${}^*\mathrm{HL}_\omega$-grammar as follows:

Construction 4. Let $HGr = \langle N, \Sigma, P, S^\circ \rangle$ be a normalized grammar. Then $\mathrm{LG}_\omega(HGr) = \langle \Sigma, S', \triangleright \rangle$ where $\triangleright$ consists of pairs $a \triangleright T_a$, and $S' = \times(S^\circ) \div \left(\sum\limits_{r \in P_N} \left({}^*_O \mathrm{DPO}(r) \right)^\bullet + \$_0^\bullet \right)$.

Here we apply the hypergraph conjunctive Kleene star to each type $\mathrm{DPO}(r)$ (for $r \in P_N$) and store the result in S'. This trick enables us to prove the following

Theorem 3. *If HGr is a normalized DPO grammar, then* $L(\mathrm{LG}_\omega(HGr)) = L(HGr)$.

Since DPO grammars generate, among others, all recursively enumerable languages, this implies:

Theorem 4. *The problem of whether a sequent is derivable in* ${}^*\mathrm{HL}_\omega$ *is undecidable.*

Acknowledgments

I thank prof. Mati Pentus and Stepan Kuznetsov for fruitful discussions, and anonymous reviewers for valuable remarks (in particular, for suggesting studying connections with process algebras).

References

[1] Frank Drewes, Hans-Jörg Kreowski & Annegret Habel (1997): *Hyperedge Replacement Graph Grammars*. In Grzegorz Rozenberg, editor: *Handbook of Graph Grammars and Computing by Graph Transformations, Volume 1: Foundations*, World Scientific, pp. 95–162, doi:10.1142/9789812384720_0002.

[2] Hartmut Ehrig, Michael Pfender & Hans Jürgen Schneider (1973): *Graph-Grammars: An Algebraic Approach*. In: *14th Annual Symposium on Switching and Automata Theory, Iowa City, Iowa, USA, October 15-17, 1973*, IEEE Computer Society, pp. 167–180, doi:10.1109/SWAT.1973.11.

[3] Makoto Kanazawa (1992): *The Lambek calculus enriched with additional connectives*. *J. Log. Lang. Inf.* 1(2), pp. 141–171, doi:10.1007/BF00171695.

[4] Barbara König, Dennis Nolte, Julia Padberg & Arend Rensink (2018): *A Tutorial on Graph Transformation*. In Reiko Heckel & Gabriele Taentzer, editors: *Graph Transformation, Specifications, and Nets - In Memory of Hartmut Ehrig*, *Lecture Notes in Computer Science* 10800, Springer, pp. 83–104, doi:10.1007/978-3-319-75396-6_5.

[5] Stepan L. Kuznetsov (2021): *Complexity of the Infinitary Lambek Calculus with Kleene Star*. *Rev. Symb. Log.* 14(4), pp. 946–972, doi:10.1017/S1755020320000209.

[6] Joachim Lambek (1958): *The Mathematics of Sentence Structure*. *The American Mathematical Monthly* 65(3), pp. 154–170, doi:10.1080/00029890.1958.11989160.

[7] Ewa Palka (2007): *An Infinitary Sequent System for the Equational Theory of *-continuous Action Lattices*. *Fundam. Informaticae* 78(2), pp. 295–309.

[8] Tikhon Pshenitsyn (2021): *Grammars Based on a Logic of Hypergraph Languages*. In Berthold Hoffmann & Mark Minas, editors: *Proceedings Twelfth International Workshop on Graph Computational Models, GCM@STAF 2021, Online, 22nd June 2021*, *EPTCS* 350, pp. 1–18, doi:10.4204/EPTCS.350.1.