Planar Realizability via Left and Right Applications

Haruka Tomita
Research Institute for Mathematical Sciences, Kyoto University, Japan

Abstract

We introduce a class of applicative structures called bi-BDI-algebras. Bi-BDI-algebras are generalizations of partial combinatory algebras and BCI-algebras, and feature two sorts of applications (left and right applications). Applying the categorical realizability construction to bi-BDI-algebras, we obtain monoidal bi-closed categories of assemblies (as well as of modest sets). We further investigate two kinds of comonadic applicative morphisms on bi-BDI-algebras as non-symmetric analogues of linear combinatory algebras, which induce models of exponential and exchange modalities on non-symmetric linear logics.

2012 ACM Subject Classification Theory of computation → Categorical semantics

Keywords and phrases Realizability, combinatory algebra, monoidal bi-closed category, exponential modality, exchange modality

Digital Object Identifier 10.4230/LIPIcs.CSL.2022.35

Funding *Haruka Tomita*: This work was supported by JST ERATO Grant Number JPMJER1603, Japan.

Acknowledgements I would like to thank Masahito Hasegawa for a lot of helpful discussions and comments. I am also grateful to Naohiko Hoshino for useful advice. Thanks also to anonymous reviewers for their helpful feedback.

1 Introduction

Categorical realizability gives us a useful method to construct categorical models of various logics and programming languages from simple structures called applicative structures. The most well known is the case of partial combinatory algebras (PCAs), which are an important class of applicative structures. From a PCA $\mathcal{A}$, we can construct Cartesian closed categories (CCCs) $\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Mod}(\mathcal{A})$ and a realizability topos $\mathbf{RT}(\mathcal{A})$ [11].

The structures of such categories obtained by realizability depend on the structures of applicative structures. Thus, assuming other conditions to applicative structures than being PCAs, we can obtain other categorical structures and use them to model other kinds of languages. A well known case is a class of applicative structure called BCI-algebras, which induces a symmetric monoidal closed structure on $\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Mod}(\mathcal{A})$ [1, 2]. While PCAs correspond to the untyped lambda calculus by the combinatory completeness property, BCI-algebras correspond to the untyped linear lambda calculus.

These two cases for PCAs and BCI-algebras are useful to give various models based on CCCs and symmetric monoidal closed categories (SMCCs). On the other hand, the categorical realizability giving rise to non-symmetric categorical structures has not been investigated much. In our previous work [14], several classes of applicative structures that induce certain non-symmetric categorical structures were introduced. The $\mathsf{BI}(-)^{\bullet}$-algebra is one of such classes. A $\mathsf{BI}(-)^{\bullet}$-algebra $\mathcal{A}$ induces the structure of closed multicategories on $\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Mod}(\mathcal{A})$, and corresponds to the untyped planar lambda calculus by combinatory completeness. Here, the planar lambda calculus is the restricted linear lambda calculus that consists of linear lambda terms whose orders of bound variables can not be freely exchanged. The name "planar" comes from the fact that planar lambda terms correspond to graphically planar maps [16, 15].

The (non-symmetric) closed multicategories obtained from $\mathsf{BI}(-)^\bullet$-algebras can be used for modeling non-symmetric implicational linear logic. However, to model richer non-symmetric logics, we additionally want to express non-symmetric tensor products and it is natural to try to get non-symmetric monoidal closed categories by categorical realizability. Unlike the symmetric case, it is a quite subtle problem to realize non-symmetric tensor products. When we try to realize tensor products by $\mathsf{BI}(-)^\bullet$-algebras in the same way as PCAs and BCI-algebras, we notice that realizers for unitors and associators induces the C-combinator and the tensor products inevitably become symmetric (and the $\mathsf{BI}(-)^\bullet$-algebra is forced to be a BCI-algebra).

This difficulty can be understood by polymorphic encoding. In $\mathbf{Asm}(\mathcal{A})$ for a BCI-algebra $\mathcal{A} = (|\mathcal{A}|, \cdot)$, realizers for an element $x_1 \otimes x_2$ of $X_1 \otimes X_2$ are $\lambda^* z.(z \cdot \mathsf{a}_1 \cdot \mathsf{a}_2)$, where a_i are realizers of x_i respectively. The form of the realizer corresponds to that $X_1 \otimes X_2$ is expressed as $\forall T.(X_1 \multimap X_2 \multimap T) \multimap T$ for symmetric cases. Whereas, for non-symmetric cases, $X_1 \otimes X_2$ is expressed as $\forall T.(T \circ\!\!- X_2 \circ\!\!- X_1) \multimap T$ and we need to distinguish two kinds of implications $\multimap$ and $\circ\!\!-$. In an applicative structure like a $\mathsf{BI}(-)^\bullet$-algebra, elements acting as functions always receive elements acting as arguments from the right side and thus the corresponding types may express only one of $\multimap$ and $\circ\!\!-$.

Conversely, if elements acting as functions can receive elements acting as arguments from both left and right side, we may construct realizers for non-symmetric tensor products. In this paper, we introduce a new structure called bi-BDI-algebra, which has two kinds of applications. These two applications correspond to $\multimap$ and $\circ\!\!-$ respectively and we can realize non-symmetric monoidal structures in $\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Mod}(\mathcal{A})$ on a bi-BDI-algebra $\mathcal{A}$.

Two applications of a bi-BDI-algebra are closely related to each other via its components $\overleftarrow{\mathsf{D}}$, $\vec{\mathsf{D}}$, $(-)^\triangleleft$ and $(-)^\triangleright$, which are introduced in order to let bi-BDI-algebras have certain combinatory completeness property. Thanks to these constructs, for a bi-BDI-algebra $\mathcal{A}$, $\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Mod}(\mathcal{A})$ consequently become monoidal bi-closed categories. Recall that monoidal bi-closed categories are monoidal categories with two kinds of adjunctions $(X \otimes -) \dashv (X \multimap -)$ and $(- \otimes Y) \dashv (- \circ\!\!- Y)$. Natural transformations relating these adjunctions indeed have realizers. In particular, the natural isomorphism $X \multimap (Y \circ\!\!- Z) \cong (X \multimap Y) \circ\!\!- Z$ is realized by $\overleftarrow{\mathsf{D}}$ and $\vec{\mathsf{D}}$.

Furthermore, by the relationship between two applications of bi-BDI-algebras, bi-BDI-algebras can be seen as a non-symmetric generalization of BCI-algebras even though bi-BDI-algebras have additional applications that BCI-algebras do not have. We can reduce a bi-BDI-algebra to a BCI-algebra by assuming certain element expressing symmetry.

In this paper, we further investigate two sorts of modalities on non-symmetric linear logics using bi-BDI-algebras. The linear exponential modality ! allows linear logics to copy and discard arguments [5]. Categorical realizability for the !-modality was introduced in [1], which uses an endomorphism on a BCI-algebra with several extra elements. In [8], the endomorphisms are generalized to total relations with certain conditions, which are generalizations of adjoint pairs between BCI-algebras and PCAs. The generalized endomorphisms give rise to comonads on categories of assemblies (or modest sets), and the comonads model !-modalities. While originally linear exponential comonads are models of !-modalities on linear logics with symmetric tensor products, later, linear exponential comonads for non-symmetric linear logics were also investigated [7]. Just as for the symmetric case, we can obtain comonads modeling !-modalities on non-symmetric linear logics by certain endomorphisms on bi-BDI-algebras. The endomorphisms are generalizations of adjoint pairs between bi-BDI-algebras and PCAs.

The exchange modality introduced in [9] allows non-symmetric linear logics to exchange arguments. A categorical model for the logic with the exchange modality is given by a monoidal adjunction between a monoidal bi-closed category and an SMCC, called a Lambek

adjoint model. We obtain such an adjunction as a co-Kleisli adjunction on categories of assemblies (or modest sets) by the similar way for !-modalities. An adjoint pair between a bi-BDI-algebra and a BCI-algebra gives rise to an endomorphism inducing a Lambek adjoint model.

The rest of this paper is structured as follows. In Section 2, we recall some basic notions and results in categorical realizability. Also we recall three classes of applicative structures: PCAs, BCI-algebras and $\mathsf{BI}(-)^{\bullet}$-algebras, which induce CCCs, SMCCs and closed multicategories respectively. In Section 3, we introduce bi-BDI-algebras and the corresponding lambda calculus. We show that bi-BDI-algebras can be seen as a generalization of BCI-algebras and that bi-BDI-algebras induce monoidal bi-closed categories. In Section 4, we construct models of linear exponential modalities and exchange modalities by categorical realizability. In Section 5, we discuss related work. Finally, in Section 6, we summarize contents of this paper.

Basic knowledge of category theory and the lambda calculus is assumed.

2 Background

2.1 Applicative structures and categories of assemblies

First we recall some basic concepts of the categorical realizability. Notations and definitions in this subsection are from [11].

▶ **Definition 1.** *A* partial applicative structure $\mathcal{A}$ *is a pair of a set* $|\mathcal{A}|$ *and a partial binary operation* $(\mathsf{x},\mathsf{y}) \mapsto \mathsf{x}\cdot\mathsf{y}$ *on* $|\mathcal{A}|$*. When the binary operation of* $\mathcal{A}$ *is total, we say* $\mathcal{A}$ *is a* total applicative structure.

Application associates to the left, and we often omit $\cdot$ and write it as juxtaposition. For instance, $\mathsf{xz(yz)}$ denotes $(\mathsf{x}\cdot\mathsf{z})\cdot(\mathsf{y}\cdot\mathsf{z})$. In the sequel, we use two notations $\downarrow$ and $\simeq$. The down arrow means "defined." For instance, for a partial applicative structure $(|\mathcal{A}|,\cdot)$, $xy\downarrow$ means that $x\cdot y$ is defined. "$\simeq$" denotes the Kleene equality, which means that if the one side of the equation is defined then the other side is also defined and they are equal.

▶ **Definition 2.** *Let* $\mathcal{A}$ *be a partial applicative structure.*

(i) *An* assembly *on* $\mathcal{A}$ *is a pair* $X := (|X|, \|\text{-}\|_X)$*, where* $|X|$ *is a set and* $\|\text{-}\|_X$ *is a function sending* $x \in |X|$ *to a non-empty subset* $\|x\|_X$ *of* $|\mathcal{A}|$*.*

(ii) *For assemblies* X *and* Y*, a* map of assemblies $f : X \to Y$ *is a function* $f : |X| \to |Y|$ *such that there exists an element* $\mathsf{r} \in |\mathcal{A}|$ *realizing* f*. Here "*r realizes f*" or "*r is a realizer of f*" means that* $\forall x \in |X|$, $\forall \mathsf{a} \in \|x\|_X$, $\mathsf{ra}\downarrow$ *and* $\mathsf{ra} \in \|f(x)\|_Y$*.*

If we assume two extra conditions on a partial applicative structure, we can construct two kinds of categories from assemblies and maps of assemblies.

▶ **Definition 3.** *Let* $\mathcal{A}$ *be a partial applicative structure satisfying that:*

1. $|\mathcal{A}|$ *has an element* I *such that for any* $\mathsf{x} \in |\mathcal{A}|$, $\mathsf{Ix}\downarrow$ *and* $\mathsf{Ix} = \mathsf{x}$*;*
2. *for any* $\mathsf{r}_1, \mathsf{r}_2 \in |\mathcal{A}|$*, there exists* $\mathsf{r}_{1,2} \in |\mathcal{A}|$ *such that for any* $\mathsf{x} \in |\mathcal{A}|$, $\mathsf{r}_{1,2}\mathsf{x} \simeq \mathsf{r}_1(\mathsf{r}_2\mathsf{x})$*.*

Then we construct categories as follows:

(i) *The* category $\mathbf{Asm}(\mathcal{A})$ of assemblies *on* $\mathcal{A}$ *consists of assemblies on* $\mathcal{A}$ *as its objects and maps of assemblies as its maps. Identity maps and composition maps are the same as those of* **Sets**.

(ii) *When an assembly* X *satisfies* $\forall x, y \in |X|$, $x \neq y \Rightarrow \|x\|_X \cap \|y\|_X = \emptyset$*, we say that* X *is a* modest set *on* $\mathcal{A}$*. The* category $\mathbf{Mod}(\mathcal{A})$ of modest sets *on* $\mathcal{A}$ *is the full subcategory of* $\mathbf{Asm}(\mathcal{A})$ *whose objects are modest sets on* $\mathcal{A}$*.*

$\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Mod}(\mathcal{A})$ are indeed categories. For any assembly $(|X|, \|\text{-}\|)$ on $\mathcal{A}$, the identity function on $|X|$ is realized by I. Given two maps of assemblies $X \xrightarrow{f} Y \xrightarrow{g} Z$ realized by r_2 and r_1 respectively, the composition function $g \circ f : |X| \to |Z|$ is realized by $\mathsf{r}_{1,2}$.

The categorical structure of $\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Mod}(\mathcal{A})$ depends on $\mathcal{A}$. When we assume some conditions on $\mathcal{A}$, $\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Mod}(\mathcal{A})$ have certain corresponding categorical structures. In the following three subsections, we introduce three classes of applicative structures: PCAs, BCI-algebras and $\mathsf{BI}(-)^\bullet$-algebras, which induce Cartesian closed categories (CCCs), symmetric monoidal closed categories (SMCCs) and closed multicategories respectively.

2.2 PCAs and Cartesian closed categories

In this subsection, we recall a well-known class of partial applicative structures called partial combinatory algebras (PCAs). PCAs correspond to the lambda calculus and assemblies on a PCA form a CCC. These results are from [11].

▶ **Definition 4.** *A* PCA *is a partial applicative structure $\mathcal{A}$ which contains two elements* S *and* K *satisfying:*

- $\forall \mathsf{x}, \mathsf{y} \in |\mathcal{A}|$, $\mathsf{Kx}\downarrow$, $\mathsf{Kxy}\downarrow$ *and* $\mathsf{Kxy} = \mathsf{x}$*;*
- $\forall \mathsf{x}, \mathsf{y}, \mathsf{z} \in |\mathcal{A}|$, $\mathsf{Sx}\downarrow$, $\mathsf{Sxy}\downarrow$ *and* $\mathsf{Sxyz} \simeq \mathsf{xz(yz)}$.

▶ **Example 5.** Suppose infinite supply of variables $x, y, z, \ldots$. *Untyped lambda terms* are terms constructed from the following six rules:

$$\frac{}{x \vdash x}\ (\text{identity}) \qquad \frac{\Gamma \vdash M \qquad \Delta \vdash N}{\Gamma, \Delta \vdash MN}\ (\text{application}) \qquad \frac{\Gamma, x \vdash M}{\Gamma \vdash \lambda x.M}\ (\text{abstraction})$$

$$\frac{\Gamma, x, y, \Delta \vdash M}{\Gamma, y, x, \Delta \vdash M}\ (\text{exchange}) \qquad \frac{\Gamma, x, y \vdash M}{\Gamma, x \vdash M[x/y]}\ (\text{contraction}) \qquad \frac{\Gamma \vdash M}{\Gamma, x \vdash M}\ (\text{weakening})$$

Here, in the application rule, Γ and Δ are sequences of distinct variables and contain no common variables. In the contraction rule, $M[x/y]$ denotes the term obtained by substituting x for all free y in M. In the weakening rule, x is a variable not contained in Γ.

Note that abstraction rules are only applied to the rightmost variables. In order to apply the abstraction rule to a variable in a different position, we need to use exchange rules and move the variable to the rightmost place.

We define *β-equivalence relation* $=_\beta$ on lambda terms as the congruence of the relation $(\lambda x.M)N \sim M[N/x]$. Untyped lambda terms modulo $=_\beta$ form a PCA. The underlying set of the PCA consists of β-equivalence classes of untyped closed lambda terms (i.e., lambda terms with no free variables) and the application is defined as that of lambda terms. In this example, $\lambda xyz.xz(yz)$ is the representative of S and $\lambda xy.x$ is the representative of K.

PCAs are closely related to the untyped lambda calculus through the property called *combinatory completeness* as in Proposition 7.

▶ **Definition 6.** *Let $\mathcal{A}$ be a partial applicative structure. A* polynominal *over $\mathcal{A}$ is a syntactic expression generated by variables, elements of $|\mathcal{A}|$ and applications. For polynominals M and N over $\mathcal{A}$, $M \simeq N$ means that $M[\mathsf{a}_1/x_1, ..., \mathsf{a}_n/x_n] \simeq N[\mathsf{a}_1/x_1, ..., \mathsf{a}_n/x_n]$ holds in $\mathcal{A}$ for any $\mathsf{a}_1, ..., \mathsf{a}_n \in |\mathcal{A}|$, where $\{x_1, ..., x_n\}$ contains all the variables of M and N.*

▶ **Proposition 7** (combinatory completeness for PCAs)**.** *Let $\mathcal{A}$ be a PCA and M be a polynominal over $|\mathcal{A}|$. For any variable x, there exists a polynominal M' such that the free variables of M' are the free variables of M excluding x and $M'\mathsf{a} \simeq M[\mathsf{a}/x]$ for all $\mathsf{a} \in |\mathcal{A}|$. We write $\lambda^* x.M$ for such M'.*

Proof. We define $\lambda^* x.M$ by induction on the structure of M as follows: $\lambda^* x.x := \mathsf{SKK}$; $\lambda^* x.y := \mathsf{K}y$ (when $x \neq y$); $\lambda^* x.MN := \mathsf{S}(\lambda^* x.M)(\lambda^* x.N)$. ◀

For the special case of the above proposition, any closed lambda term is β-equivalent to some term constructed from $\lambda xyz.xz(yz)$ and $\lambda xy.x$ only using applications.

Although conditions of PCAs are simple, categorical structures induced by PCAs are quite strong and useful.

▶ **Proposition 8.** *Let $\mathcal{A}$ be a PCA. Then $\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Mod}(\mathcal{A})$ are Cartesian closed and regular.*

The detailed proof is in [11]. What is important here is how to realize products and exponents by PCAs. Cartesian closed structures in $\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Mod}(\mathcal{A})$ are defined as follows. The terminal object is $(\{*\}, \|\text{-}\|_1)$, where $\|*\|_1 := |\mathcal{A}|$. For objects X and Y, the product is $(|X| \times |Y|, \|\text{-}\|_{X\times Y})$, where $\|(x,y)\|_{X\times Y} := \{\lambda^* z.z\mathsf{a}\mathsf{a}' \mid \mathsf{a} \in \|x\|_X,\ \mathsf{a}' \in \|y\|_Y\}$. Projection maps are realized by K and $\lambda^* xy.y$. For objects X and Y, the exponent is defined as $(Hom_{\mathbf{Asm}(\mathcal{A})}(X,Y), \|\text{-}\|_{Y^X})$, where $\|f\|_{Y^X} := \{\mathsf{r} \mid \mathsf{r} \text{ realizes } f\}$. The evaluation map $ev : X \times Y^X \to Y$ is realized by $\lambda^* z.z(\lambda^* uv.vu)$.

2.3 BCI-algebras and symmetric monoidal closed categories

In this subsection we recall another class of applicative structures called BCI-algebra. BCI-algebras are related to linear structures whereas PCAs are not. The results given below are from [2, 8].

▶ **Definition 9.** *A* BCI-algebra *is a total applicative structure $\mathcal{A}$ which contains three elements* B, C *and* I *such that for any* $\mathsf{x},\mathsf{y},\mathsf{z} \in |\mathcal{A}|$, $\mathsf{Bxyz} = \mathsf{x}(\mathsf{yz})$, $\mathsf{Cxyz} = \mathsf{xzy}$ *and* $\mathsf{Ix} = \mathsf{x}$.

▶ **Example 10.** *Untyped linear lambda terms* are untyped lambda terms constructed without using weakening and contraction rules, i.e., untyped lambda terms whose each variable appears just once. Untyped linear lambda terms modulo $=_\beta$ form a BCI-algebra. Here $\lambda xyz.x(yz)$, $\lambda xyz.xzy$ and $\lambda x.x$ are the representatives of B, C and I respectively.

▶ **Proposition 11** (combinatory completeness for BCI-algebras)**.** *Let $\mathcal{A}$ be a* BCI*-algebra and M be a polynominal over $|\mathcal{A}|$. For any variable x appearing exactly once in M, there exists a polynominal $\lambda^* x.M$ such that the free variables of $\lambda^* x.M$ are the free variables of M excluding x and $(\lambda^* x.M)\mathsf{a} = M[\mathsf{a}/x]$ for all $\mathsf{a} \in |\mathcal{A}|$.*

Proof. We define $\lambda^* x.M$ by induction on the structure of M as follows:

- $\lambda^* x.x := \mathsf{I}$
- $\lambda^* x.MN := \begin{cases} \mathsf{C}(\lambda^* x.M)N & (x \in FV(M)) \\ \mathsf{B}M(\lambda^* x.N) & (x \in FV(N)) \end{cases}$ ◀

For the special case of the above proposition, any closed linear lambda term is β-equivalent to some term constructed from $\lambda xyz.x(yz)$, $\lambda xyz.xzy$ and $\lambda x.x$ only using applications.

Since BCI-algebras are related to the linear lambda calculus, categorical structures of assemblies on BCI-algebras are also linear ones.

▶ **Proposition 12.** *Let $\mathcal{A}$ be a* BCI*-algebra. Then $\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Mod}(\mathcal{A})$ are SMCCs.*

The monoidal structure in $\mathbf{Asm}(\mathcal{A})$ are defined as follows: The unit object is $(\{*\}, \|\text{-}\|_I)$, where $\|*\|_I := \{\mathsf{I}\}$. For objects X and Y, the tensor product is defined as $(|X| \times |Y|, \|\text{-}\|_{X\otimes Y})$, where $\|(x,y)\|_{X\otimes Y} := \{\lambda^* z.z\mathsf{a}\mathsf{a}' \mid \mathsf{a} \in \|x\|_X,\ \mathsf{a}' \in \|y\|_Y\}$. Realizers for natural isomorphisms

follows by the combinatory completeness. For instance, the associator $a : (X \otimes Y) \otimes Z \to X \otimes (Y \otimes Z)$ is realized by $\lambda^* w.w(\lambda^* w' r.w'(\lambda^* pq.(\lambda^* u.up(\lambda^* v.vqr))))$. The monoidal structures of $\mathbf{Mod}(\mathcal{A})$ are not the same as $\mathbf{Asm}(\mathcal{A})$, since $X \otimes Y$ may not be a modest set even if X and Y are modest. We can define the monoidal structures of $\mathbf{Mod}(\mathcal{A})$ by the monoidal adjunction associated with $\mathbf{Mod}(\mathcal{A})$ being a reflective fullsubcategory of $\mathbf{Asm}(\mathcal{A})$. (See Section 3 in [8].)

2.4 $\mathsf{BI}(-)^\bullet$-algebras and closed multicategories

In this subsection we give another class of applicative structure called $\mathsf{BI}(-)^\bullet$-algebras. They are generalizations of BCI-algebras by excluding C that expresses exchanging arguments. $\mathsf{BI}(-)^\bullet$-algebras are related to non-symmetric linear structures. These results are from [14].

▶ **Definition 13.** *We say that a total applicative structure $\mathcal{A}$ is a* $\mathsf{BI}(-)^\bullet$-algebra *iff it contains* B, I *and* $\mathsf{a}^\bullet$ *for each* $\mathsf{a} \in |\mathcal{A}|$, *where* $\mathsf{a}^\bullet$ *is an element of* $|\mathcal{A}|$ *such that* $\mathsf{a}^\bullet \mathsf{x} = \mathsf{xa}$ *for all* $\mathsf{x} \in |\mathcal{A}|$.

Like PCAs and BCI-algebras, $\mathsf{BI}(-)^\bullet$-algebras are related to a part of the untyped lambda calculus, which is called the *planar lambda calculus*.

▶ **Example 14.** *Untyped planar lambda terms* are untyped lambda terms constructed without using weakening, contraction nor exchange rules. Untyped closed planar lambda terms modulo $=_\beta$ form a $\mathsf{BI}(-)^\bullet$-algebra. Here $\lambda xyz.x(yz)$ and $\lambda x.x$ are the representatives of B and I. Given a representative M of a, $\lambda x.xM$ is also a closed planar term and is the representative of $\mathsf{a}^\bullet$.

▶ **Proposition 15** (combinatory completeness for $\mathsf{BI}(-)^\bullet$-algebras)**.** *Let $\mathcal{A}$ be a* $\mathsf{BI}(-)^\bullet$*-algebra and M be a polynominal over $|\mathcal{A}|$. For the rightmost variable x, which has to appear exactly once in M, there exists a polynominal $\lambda^* x.M$ such that the free variables of $\lambda^* x.M$ are the free variables of M in the same order excluding x and $(\lambda^* x.M)\mathsf{a} = M[\mathsf{a}/x]$ for all $\mathsf{a} \in |\mathcal{A}|$.*

Proof. We define $\lambda^* x.M$ by induction on the structure of M as follows:

- $\lambda^* x.x := \mathsf{I}$
- $\lambda^* x.MN := \begin{cases} \mathsf{B}N^\bullet(\lambda^* x.M) & (x \in FV(M)) \\ \mathsf{B}M(\lambda^* x.N) & (x \in FV(N)) \end{cases}$

Note that for $\lambda^* x.MN$, x is the rightmost free variable in MN. Therefore, if x is in $FV(M)$, N has no free variables and $N^\bullet$ can be defined. ◀

For the special case of the above proposition, any closed planar lambda term is β-equivalent to some term constructed from $\lambda xyz.x(yz)$ and $\lambda x.x$ using applications and the unary operation $(-)^\bullet : M \mapsto \lambda x.xM$.

$\mathsf{BI}(-)^\bullet$-algebras are related to the planar lambda calculus, that is, a fragment of the linear lambda calculus without symmetry. Therefore, categorical structures of assemblies on $\mathsf{BI}(-)^\bullet$-algebras also are non-symmetric and linear.

▶ **Proposition 16.** *For a* $\mathsf{BI}(-)^\bullet$*-algebra $\mathcal{A}$,* $\mathbf{Asm}(\mathcal{A})$ *and* $\mathbf{Mod}(\mathcal{A})$ *are closed multicategories.*

Here closed multicategories are one of generalizations of categories with objects expressing internal hom [12]. What we need to refer here is that closed multicategories are generalization of monoidal closed categories and they not generally have tensors. When we try to construct realizers for tensor products in $\mathbf{Asm}(\mathcal{A})$ as we did in Proposition 8 and Proposition 12 ($\|x \otimes y\| := \{\lambda^* z.z\mathsf{aa}' \mid \mathsf{a} \in \|x\|, \mathsf{a}' \in \|y\|\}$), we notice that realizers for associators and unitors do not generally exist. Even if we assume these realizers in a $\mathsf{BI}(-)^\bullet$-algebra, we can

construct an element acting as C from these realizers. Take an assembly X as $|X| := |\mathcal{A}|$ and $\|\mathsf{a}\|_X := \{\mathsf{a}\}$. Assuming that the unitor $X \to I \otimes X$ has a realizer r, $\mathsf{rxz} = \mathsf{zIx}$ holds for any x and z. Let $C' := \lambda^* xz.\mathsf{r}x(\mathsf{B}z)$ and $C := \lambda^* xy.C'x(\mathsf{B}(C'y))$. Then C satisfies the axiom of the C-combinator and thus $\mathcal{A}$ inevitably becomes a BCI-algebra.

2.5 Applicative morphisms

In this subsection, we recall the notion of applicative morphisms from [11]. In [11], applicative morphisms are defined not for arbitrary applicative structures but only for PCAs. However, the same definition makes sense for a large class of applicative structures including PCAs, BCI-algebras and $\mathsf{BI}(-)^\bullet$-algebras.

In this subsection, let $\mathcal{A}$ and $\mathcal{B}$ range over PCAs and $\mathsf{BI}(-)^\bullet$-algebras.

▶ **Definition 17.** *An* applicative morphism $\gamma : \mathcal{A} \to \mathcal{B}$ *is a total relation from* $|\mathcal{A}|$ *to* $|\mathcal{B}|$ *such that there is a* realizer $\mathsf{r}_\gamma \in |\mathcal{B}|$ *of* γ *satisfying that for any* $\mathsf{x}, \mathsf{x}' \in |\mathcal{A}|$, $\mathsf{y} \in \gamma\mathsf{x}$ *and* $\mathsf{y}' \in \gamma\mathsf{x}'$, $\mathsf{r}_\gamma\mathsf{y}\mathsf{y}' \in \gamma(\mathsf{x}\mathsf{x}')$ *holds whenever* $\mathsf{x}\mathsf{x}' \downarrow$. *(We often write such a condition as* $\mathsf{r}_\gamma(\gamma\mathsf{x})(\gamma\mathsf{x}') \subseteq \gamma(\mathsf{x}\mathsf{x}')$*).*

▶ **Definition 18.** *For two applicative morphisms* $\gamma, \delta : \mathcal{A} \to \mathcal{B}$, $\gamma \preceq \delta$ *iff there exists an element* $\mathsf{r} \in |\mathcal{B}|$ *called* realizer *of* $\gamma \preceq \delta$ *such that* $\mathsf{ry} \in \delta\mathsf{x}$ *for any* $\mathsf{x} \in |\mathcal{A}|$ *and* $\mathsf{y} \in \gamma\mathsf{x}$.

By the preorder $\preceq$, we can define adjunctions and comonads on applicative structures.

▶ **Definition 19.** *For applicative morphisms* $\gamma : \mathcal{A} \to \mathcal{B}$ *and* $\delta : \mathcal{B} \to \mathcal{A}$, γ *is a* right adjoint *of* δ *iff* $\delta \circ \gamma \preceq id_{\mathcal{A}}$ *and* $id_{\mathcal{B}} \preceq \gamma \circ \delta$. *We often write these settings as* $(\delta \dashv \gamma) : \mathcal{A} \to \mathcal{B}$.

▶ **Definition 20.** *We say an applicative morphism* $\gamma : \mathcal{A} \to \mathcal{A}$ *is a* comonadic applicative morphism *when* $\mathcal{A}$ *has two element* e *and* d *such that* $\mathsf{e}(\gamma\mathsf{x}) \subseteq \{\mathsf{x}\}$ *and* $\mathsf{d}(\gamma\mathsf{x}) \subseteq \gamma(\gamma\mathsf{x})$ *for any* $\mathsf{x} \in |\mathcal{A}|$.

Given an adjoint pair of applicative morphisms $(\delta \dashv \gamma) : \mathcal{A} \to \mathcal{B}$, we obtain a comonadic applicative morphism $(\delta \circ \gamma) : \mathcal{A} \to \mathcal{A}$. e is given as a realizer of $\delta \circ \gamma \preceq id_{\mathcal{A}}$ and d is given as $\mathsf{r}_\delta(\delta\mathsf{r})$, where r_δ is a realizer of δ and r is a realizer of $id_{\mathcal{B}} \preceq \gamma \circ \delta$.

Any applicative morphism $\gamma : \mathcal{A} \to \mathcal{B}$ gives rise to a functor $\gamma_* : \mathbf{Asm}(\mathcal{A}) \to \mathbf{Asm}(\mathcal{B})$. Furthermore, any adjoint pair $(\delta \dashv \gamma) : \mathcal{A} \to \mathcal{B}$ gives rise to an adjunction $\delta_* \dashv \gamma_*$.

▶ **Definition 21.** *For an applicative morphism* $\gamma : \mathcal{A} \to \mathcal{B}$, $\gamma_* : \mathbf{Asm}(\mathcal{A}) \to \mathbf{Asm}(\mathcal{B})$ *is the functor sending an object* $(|X|, \|\text{-}\|_X)$ *to* $(|X|, \gamma\|\text{-}\|_X)$ *and sending a map* $f : X \to Y$ *to the same function.*

The realizer of $\gamma_* f$ exists in $\mathsf{r}_\gamma(\gamma\mathsf{r}_f)$, where r_γ and r_f are realizers of γ and f respectively.

▶ **Proposition 22.** *An adjoint pair of applicative morphisms* $(\delta \dashv \gamma) : \mathcal{A} \to \mathcal{B}$ *gives rise to an adjunction* $\delta_* \dashv \gamma_* : \mathbf{Asm}(\mathcal{A}) \to \mathbf{Asm}(\mathcal{B})$.

The unit and counit are realized by the realizers of $id_{\mathcal{B}} \preceq \gamma \circ \delta$ and $\delta \circ \gamma \preceq id_{\mathcal{A}}$ respectively.

For a comonadic applicative morphism $\gamma : \mathcal{A} \to \mathcal{A}$, γ_* is a comonad on $\mathbf{Asm}(\mathcal{A})$. The counit is realized by e and the comultiplication is realized by d.

This subsection can be summarized as follows. PCAs and $\mathsf{BI}(-)^\bullet$-algebras form a preorder enriched category and $\mathbf{Asm}(-)$ extends to a 2-functor from this 2-category to the 2-category of categories of assemblies (on PCAs and $\mathsf{BI}(-)^\bullet$-algebras). Details for the 2-functor $\mathbf{Asm}(-)$ (for PCAs) are in Section 2.2 of [11].

▶ Remark 23. Given an applicative morphism $\gamma : \mathcal{A} \to \mathcal{B}$, we can not generally obtain a functor $\gamma_* : \mathbf{Mod}(\mathcal{A}) \to \mathbf{Mod}(\mathcal{B})$ as the same for categories of assemblies, since $\|x\|_X \cap \|x'\|_X = \emptyset$ does not imply $\gamma(\|x\|_X) \cap \gamma(\|x'\|_X) = \emptyset$ and $\gamma_* X$ may not be in $\mathbf{Mod}(\mathcal{B})$. However, for a comocadic applicative morphism $\gamma : \mathcal{A} \to \mathcal{A}$, γ_* can be restricted to an endofunctor on $\mathbf{Mod}(\mathcal{A})$. Indeed, for a modest set X, if $\mathsf{a} \in (\gamma\|x\|_X) \cap (\gamma\|x'\|_X)$ then $\mathsf{ea} \in \|x\|_X \cap \|x'\|_X$ and thus $x = x'$. Just like for $\mathbf{Asm}(\mathcal{A})$, the γ_* is a comonad on $\mathbf{Mod}(\mathcal{A})$.

3 Bi-BDI-algebras and monoidal bi-closed categories

As we said in Section 2.4, it is difficult to construct non-symmetric monoidal structure on $\mathbf{Asm}(\mathcal{A})$ by $\mathsf{BI}(-)^\bullet$-algebras in the same way as BCI-algebras and PCAs. We need some major modification on the definition of realizers of tensor products in $\mathbf{Asm}(\mathcal{A})$.

Here we introduce a new class of applicative structures called bi-BDI-algebras, which are very different from classes of applicative structures we have seen so far since bi-BDI-algebras contain two sorts of applications. We use the two applications to realize tensor products while avoiding intrusion of C-combinators.

3.1 Bi-BDI-algebras and the bi-planar lambda calculus

First we introduce a variant of the lambda calculus that we call the *bi-planar lambda calculus* here, which contains two sides of applications and abstractions[1].

▶ **Definition 24.** Bi-planar lambda terms *are constructed by the following rules:*

$$\frac{}{x \vdash x}\ \textit{(identity)} \qquad \frac{\Gamma, x \vdash M}{\Gamma \vdash (M \leftarrow\!\!\shortmid x)}\ \textit{(right abstraction)} \qquad \frac{x, \Gamma \vdash M}{\Gamma \vdash (x \mapsto\!\!\shortmid M)}\ \textit{(left abstraction)}$$

$$\frac{\Gamma \vdash M \qquad \Delta \vdash N}{\Gamma, \Delta \vdash M \mathbin{\overleftarrow{@}} N}\ \textit{(right application)} \qquad \frac{\Gamma \vdash M \qquad \Delta \vdash N}{\Gamma, \Delta \vdash M \mathbin{\vec{@}} N}\ \textit{(left application)}$$

Here is none of weakening, contraction nor exchange rules.

Although bi-planar lambda terms seem very different from ordinary lambda terms, when we construct terms only using identity, right application and right abstraction rules, they are planar lambda terms. In this case $M \mathbin{\overleftarrow{@}} N$ denotes MN and $(M \leftarrow\!\!\shortmid x)$ denotes $\lambda x.M$.

For the sake of clarity, we classify right and left by red and blue. That is, we write each of them as $M \mathbin{\overleftarrow{@}} N$, $(M \leftarrow\!\!\shortmid x)$, $N \mathbin{\vec{@}} M$ and $(x \mapsto\!\!\shortmid M)$.

▶ **Definition 25.** *We define a relation $\to_\beta$ on bi-planar lambda terms as the congruence of the following relations:*

- *(right β-reduction)* $(M \leftarrow\!\!\shortmid x) \mathbin{\overleftarrow{@}} N \to_\beta M[N/x]$
- *(left β-reduction)* $N \mathbin{\vec{@}} (x \mapsto\!\!\shortmid M) \to_\beta M[N/x]$

The bi-planar lambda calculus *consists of bi-planar lambda terms and the reflexive, symmetric and transitive closure of $\to_\beta$ as the equational relation $=_\beta$.*

Basic properties about $\to_\beta$, such as the confluence and the strongly normalizing property, can be shown in the same way as the proof for the linear lambda calculus.

[1] The terminology left and right abstractions corresponds to left and right closed structures of monoidal categories: $(X \otimes -) \dashv (X \multimap -)$ and $(- \otimes Y) \dashv (- \circ\!\!- Y)$.

▶ Remark 26. The bi-planar lambda calculus is essentially not a new concept, since it often appears as the Curry-Howard corresponding calculus with the Lambek calculus (cf. [9]). However, note that unlike the calculus corresponding to the Lambek calculus, the bi-planar lambda calculus is based on untyped setting. The reason why we use a less-standard notation is to shorten the length of the realizers and to make them easier to read.

Next we introduce the notion of bi-BDI-algebra, which corresponds to the bi-planar lambda calculus. In order to express the bi-planar lambda calculus by an algebraic structure, the structure is not enough to have two sides of applications, but also need some conditions for relating these two applications. In bi-BDI-algebras, $\overleftarrow{\mathsf{D}}$, $\vec{\mathsf{D}}$, $(-)^{\triangleleft}$ and $(-)^{\triangleright}$ express such conditions.

▶ **Definition 27.** *We say a total applicative structure* $\mathcal{A} = (|\mathcal{A}|, \overleftarrow{@})$ *is a* bi-BDI-algebra *when there is an additional total binary operation* $\vec{@}$ *on* $|\mathcal{A}|$ *and* $|\mathcal{A}|$ *contains several special elements:*

- $\overleftarrow{\mathsf{B}} \in |\mathcal{A}|$ *such that* $((\overleftarrow{\mathsf{B}} \overleftarrow{@} \mathsf{x}) \overleftarrow{@} \mathsf{y}) \overleftarrow{@} \mathsf{z} = \mathsf{x} \overleftarrow{@} (\mathsf{y} \overleftarrow{@} \mathsf{z})$ *for any* $\mathsf{x}, \mathsf{y}, \mathsf{z} \in |\mathcal{A}|$.
- $\vec{\mathsf{B}} \in |\mathcal{A}|$ *such that* $\mathsf{z} \vec{@} (\mathsf{y} \vec{@} (\mathsf{x} \vec{@} \vec{\mathsf{B}})) = (\mathsf{z} \vec{@} \mathsf{y}) \vec{@} \mathsf{x}$ *for any* $\mathsf{x}, \mathsf{y}, \mathsf{z} \in |\mathcal{A}|$.
- $\overleftarrow{\mathsf{D}} \in |\mathcal{A}|$ *such that* $\mathsf{x} \vec{@} ((\overleftarrow{\mathsf{D}} \overleftarrow{@} \mathsf{y}) \overleftarrow{@} \mathsf{z}) = (\mathsf{x} \vec{@} \mathsf{y}) \overleftarrow{@} \mathsf{z}$ *for any* $\mathsf{x}, \mathsf{y}, \mathsf{z} \in |\mathcal{A}|$.
- $\vec{\mathsf{D}} \in |\mathcal{A}|$ *such that* $(\mathsf{z} \vec{@} (\mathsf{y} \vec{@} \vec{\mathsf{D}})) \overleftarrow{@} \mathsf{x} = \mathsf{z} \vec{@} (\mathsf{y} \overleftarrow{@} \mathsf{x})$ *for any* $\mathsf{x}, \mathsf{y}, \mathsf{z} \in |\mathcal{A}|$.
- $\overleftarrow{\mathsf{I}} \in |\mathcal{A}|$ *such that* $\overleftarrow{\mathsf{I}} \overleftarrow{@} \mathsf{x} = \mathsf{x}$ *for any* $\mathsf{x} \in |\mathcal{A}|$.
- $\vec{\mathsf{I}} \in |\mathcal{A}|$ *such that* $\mathsf{x} \vec{@} \vec{\mathsf{I}} = \mathsf{x}$ *for any* $\mathsf{x} \in |\mathcal{A}|$.
- *For each* $\mathsf{a} \in |\mathcal{A}|$, $\mathsf{a}^{\triangleleft} \in |\mathcal{A}|$ *such that* $(\mathsf{a}^{\triangleleft}) \overleftarrow{@} \mathsf{x} = \mathsf{x} \vec{@} \mathsf{a}$ *for any* $\mathsf{x} \in |\mathcal{A}|$.
- *For each* $\mathsf{a} \in |\mathcal{A}|$, $\mathsf{a}^{\triangleright} \in |\mathcal{A}|$ *such that* $\mathsf{x} \vec{@} (\mathsf{a}^{\triangleright}) = \mathsf{a} \overleftarrow{@} \mathsf{x}$ *for any* $\mathsf{x} \in |\mathcal{A}|$.

We call $\overleftarrow{@}$ *and* $\vec{@}$ *as* right application *and* left application *respectively. In the sequel, we use* $\overleftarrow{@}$ *as a left-associative operation and often omit unnecessary parentheses, while we do not omit parentheses for* $\vec{@}$.

As can be seen from the definition, though the left application is an extra component in a bi-BDI-algebra, the conditions required for left and right applications are dual. We often write $\mathcal{A} = (|\mathcal{A}|, \overleftarrow{@}, \vec{@})$ as a bi-BDI-algebra $\mathcal{A} = (|\mathcal{A}|, \overleftarrow{@})$ with the left application $\vec{@}$.

▶ Remark 28. Here we deal with bi-BDI-algebras only as total applicative structures while we can define *partial* bi-BDI-algebras. Given a partial bi-BDI-algebra $\mathcal{A}$, we always can extend $\mathcal{A}$ to a total bi-BDI-algebra $\mathcal{A}'$ by adding an extra element expressing "undefined." Then $\mathbf{Asm}(\mathcal{A})$ is a full subcategory of $\mathbf{Asm}(\mathcal{A}')$. The same discussion for partial BCI-algebras is in Remark 1 of [8].

▶ **Example 29.** Closed bi-planar lambda terms modulo $=_\beta$ form a bi-BDI-algebra. For instance, $(((x \overleftarrow{@} (y \overleftarrow{@} z)) \leftarrow\!\shortmid z) \leftarrow\!\shortmid y) \leftarrow\!\shortmid x)$, $(((x \mapsto (x \vec{@} y) \overleftarrow{@} z) \leftarrow\!\shortmid z) \leftarrow\!\shortmid y)$ and $(x \mapsto M \overleftarrow{@} x)$ are the representative of $\overleftarrow{\mathsf{B}}$, $\overleftarrow{\mathsf{D}}$ and $M^{\triangleright}$.

Combinatory completeness holds for bi-BDI-algebras and the bi-planar lambda calculus.

▶ **Proposition 30** (Combinatory completeness for bi-BDI-algebras)**.** *Let* $\mathcal{A}$ *be a bi-*BDI*-algebra. We define a* polynominal over $\mathcal{A}$ *as a syntactic expression generated by variables, elements of* $|\mathcal{A}|$ *and left and right applications of* $\mathcal{A}$*. Suppose a polynominal* M *over* $\mathcal{A}$ *and the rightmost variable* x *appears only once in* M*. There exists a polynominal* M' *such that the free variables of* M' *are the free variables of* M *excluding* x *and* $M' \overleftarrow{@} \mathsf{a} = M'[\mathsf{a}/x]$ *for any* $\mathsf{a} \in |\mathcal{A}|$*. We write* $(M \leftarrow\!\shortmid x)$ *for such* M'*. Also, if the leftmost variable* x *appears only once in* M*, there exists a polynominal* M'' *such that the free variables of* M'' *are the free variables of* M *excluding* x *and* $\mathsf{a} \vec{@} M'' = M''[\mathsf{a}/x]$ *for any* $\mathsf{a} \in |\mathcal{A}|$*. We write* $(x \mapsto M)$ *for such* M''.

Proof. We define $(x{\mapsto}M)$ by induction on the structure of M.

- $(x{\mapsto}x) := \vec{\mathsf{I}}$
- $(x{\mapsto}(N \mathbin{\vec{@}} M)) := \begin{cases} (x{\mapsto}M) \mathbin{\vec{@}} ((N \mathbin{\vec{@}} (\overleftarrow{\mathsf{I}} \mathbin{\vec{@}} \vec{\mathsf{D}}))^{\triangleright} \mathbin{\vec{@}} \vec{\mathsf{B}}) & (x \in FV(M)) \\ (x{\mapsto}N) \mathbin{\vec{@}} (M \mathbin{\vec{@}} \vec{\mathsf{B}}) & (x \in FV(N)) \end{cases}$

  For the case $x \in FV(M)$, x is the leftmost free variable of $N \mathbin{\vec{@}} M$. Hence N contains no variables and $(N \mathbin{\vec{@}} (\overleftarrow{\mathsf{I}} \mathbin{\vec{@}} \vec{\mathsf{D}}))^{\triangleright}$ can be defined.
- $(x{\mapsto}(M \mathbin{\overleftarrow{@}} N)) := \begin{cases} (\overleftarrow{\mathsf{D}} \mathbin{\overleftarrow{@}} (x{\mapsto}M)) \mathbin{\overleftarrow{@}} N & (x \in FV(M)) \\ (x{\mapsto}N) \mathbin{\vec{@}} ((M^{\triangleright}) \mathbin{\vec{@}} \vec{\mathsf{B}}) & (x \in FV(N)) \end{cases}$

  For the case $x \in FV(N)$, x is the leftmost free variable of $M \mathbin{\overleftarrow{@}} N$. Hence M contains no variables and $M^{\triangleright}$ can be defined.

The case of the right abstractions $(M{\leftarrow}x)$ is given in the same way, with all the left and right constructs exchanged. ◀

It immediately follows that for any bi-BDI-algebra $(|\mathcal{A}|, \overleftarrow{@}, \vec{@})$, the applicative structure $(|\mathcal{A}|, \overleftarrow{@})$ is a $\mathsf{BI}(-)^{\bullet}$-algebra since $\mathsf{a}^{\bullet}$ can be defined as $(x \mathbin{\overleftarrow{@}} \mathsf{a}{\leftarrow}x)$.

We can show that the left application of a bi-BDI-algebra is unique up to isomorphism.

▶ **Proposition 31.** *Suppose that $\mathcal{A}_1 = (|\mathcal{A}|, \overleftarrow{@}, \vec{@}_1)$ and $\mathcal{A}_2 = (|\mathcal{A}|, \overleftarrow{@}, \vec{@}_2)$ are bi-BDI-algebras. Then $(|\mathcal{A}|, \overleftarrow{@}_1)$ and $(|\mathcal{A}|, \overleftarrow{@}_2)$ are isomorphic as applicative structures, where $x \mathbin{\overleftarrow{@}_i} y := y \mathbin{\vec{@}_i} x$.*

To the end of this subsection, we additionally give an example of a bi-BDI-algebra.

▶ **Example 32.** Take an ordered group $(G, \cdot, e, \leq)$. Let T be the set of terms t constructed as follows:

$$t ::= g \mid t \circ\!\!- t \mid t -\!\!\circ t \quad (g \in G).$$

We define a function $|\text{-}| : T \to G$ by induction: $|g| := g$, $|t_1 \circ\!\!- t_2| := |t_1| \cdot |t_2|^{-1}$ and $|t_2 -\!\!\circ t_1| := |t_2|^{-1} \cdot |t_1|$.

Let $\mathcal{T}$ be the powerset of $\{t \in T \mid e \leq |t|\}$. Then $\mathcal{T}$ forms a bi-BDI-algebra.

- For $M, N \in \mathcal{T}$, $M \mathbin{\overleftarrow{@}} N := \{t_1 \mid \exists t_2 \in N, (t_1 \circ\!\!- t_2) \in M\}$.
- For $M, N \in \mathcal{T}$, $N \mathbin{\vec{@}} M := \{t_1 \mid \exists t_2 \in N, (t_2 -\!\!\circ t_1) \in M\}$.
- $\overleftarrow{\mathsf{B}} := \{((t_1 \circ\!\!- t_3) \circ\!\!- (t_2 \circ\!\!- t_3)) \circ\!\!- (t_1 \circ\!\!- t_2) \mid t_1, t_2, t_3 \in T\}$, dual for $\vec{\mathsf{B}}$.
- $\overleftarrow{\mathsf{D}} := \{((t_1 -\!\!\circ t_2) \circ\!\!- t_3) \circ\!\!- (t_1 -\!\!\circ (t_2 \circ\!\!- t_3)) \mid t_1, t_2, t_3 \in T\}$, dual for $\vec{\mathsf{D}}$.
- $\overleftarrow{\mathsf{I}} := \{t_1 -\!\!\circ t_1 \mid t_1 \in T\}$, same for $\vec{\mathsf{I}}$.
- For $M \in \mathcal{T}$, $M^{\triangleleft} := \{t_1 \circ\!\!- t_2 \mid (t_2 -\!\!\circ t_1) \in M\}$, same for $M^{\triangleright}$.

The construction of this example is the same one as that for a $\mathsf{BI}(-)^{\bullet}$-algebra in Section 6 of [14], which is based on a reflexive object of a pivotal category $\mathcal{C}omod(\overline{G})$ introduced in [6].

3.2 Bi-BDI-algebras and BCI-algebras

In this subsection, we show that bi-BDI-algebras can be seen as non-commutative generalizations of BCI-algebras. First, we show that the BCI-algebra is a special case of the bi-BDI-algebra.

▶ **Proposition 33.** *Let $\mathcal{B} = (|\mathcal{B}|, \cdot)$ be a BCI-algebra. When we take two binary operations $\vec{@}$ and $\overleftarrow{@}$ by $y \mathbin{\vec{@}} x = x \mathbin{\overleftarrow{@}} y := x \cdot y$, $(|\mathcal{B}|, \overleftarrow{@}, \vec{@})$ is a bi-BDI-algebra.*

Proof. B satisfies axioms for $\vec{\mathsf{B}}$ and $\overleftarrow{\mathsf{B}}$. I satisfies axioms for $\vec{\mathsf{I}}$ and $\overleftarrow{\mathsf{I}}$. C satisfies axioms for $\overleftarrow{\mathsf{D}}$ and $\vec{\mathsf{D}}$. x satisfies axioms for $\mathsf{x}^{\triangleright}$ and $\mathsf{x}^{\triangleleft}$. ◀

Conversely, not all bi-BDI-algebras are BCI-algebras. Indeed, the bi-planar lambda calculus is not a BCI-algebra. Here we show that when one of the left/right applicative structures of a bi-BDI-algebra is a BCI-algebra, so is the other applicative structure and moreover they coincide.

▶ **Proposition 34.** *Let $\mathcal{A} = (|\mathcal{A}|, \overleftarrow{@}, \vec{@})$ be a bi-*BDI*-algebra. Take an applicative structures $\mathcal{A}' = (|\mathcal{A}|, \overleftarrow{@}')$ by $\mathsf{x} \overleftarrow{@}' \mathsf{y} := \mathsf{y} \vec{@} \mathsf{x}$. Then $\mathcal{A}$ is a* BCI*-algebra iff $\mathcal{A}'$ is a* BCI*-algebra. Moreover, in such a case, these* BCI*-algebras are isomorphic as applicative structures.*

3.3 Bi-BDI-algebras and monoidal bi-closed categories

Now we show the main result that bi-BDI-algebras induce monoidal bi-closed categories. For a bi-BDI-algebra $\mathcal{A} = (|\mathcal{A}|, \overleftarrow{@}, \vec{@})$, $\mathbf{Asm}(\mathcal{A})$ is constructed by considering $\mathcal{A}$ as an applicative structure $(|\mathcal{A}|, \overleftarrow{@})$.

▶ **Proposition 35.** *For a bi-*BDI*-algebra $\mathcal{A}$, $\mathbf{Asm}(\mathcal{A})$ is a monoidal bi-closed category.*

Proof. (Sketch)

- For objects X and Y, the underlying set of $X \otimes Y$ is $|X| \times |Y|$. Realizers are defined as $\|x \otimes y\| := \{(z \mapsto z \overleftarrow{@} \mathsf{a} \overleftarrow{@} \mathsf{a}') \mid \mathsf{a} \in \|x\|_X, \mathsf{a}' \in \|y\|_Y\}$.
- For $f : X \to X'$ and $g : Y \to Y'$, the map $f \otimes g$ is a function sending $x \otimes y$ to $f(x) \otimes g(y)$. The realizer for $f \otimes g$ is $((((z \mapsto z \overleftarrow{@} (\mathsf{r}_f \overleftarrow{@} p) \overleftarrow{@} (\mathsf{r}_g \overleftarrow{@} q)) \leftarrow q) \leftarrow p) \vec{@} w \leftarrow w)$.
- The underlying set of the unit object I is a singleton $\{*\}$. The realizer is $\|*\|_I := \{\overleftarrow{\mathsf{I}}\}$.
- The left unitor $l : I \otimes X \to X$ sends $* \otimes x$ to x, whose realizer is $(\overleftarrow{\mathsf{I}} \vec{@} w \leftarrow w)$. The realizer of l^{-1} is $((z \mapsto z \overleftarrow{@} \overleftarrow{\mathsf{I}} \overleftarrow{@} x) \leftarrow x)$.
- For objects X and Y, the underlying set of $X \multimap Y$ is the set of maps from X to Y. $\|f\| := \{\mathsf{r} \in |\mathcal{A}| \mid \mathsf{a} \vec{@} \mathsf{r} \in \|f(x)\|_Y$ for any $x \in |X|$ and $\mathsf{a} \in \|x\|_X\}$. This set is not empty since $(\mathsf{r}_f)^{\triangleright}$ is in the set for a realizer r_f of f.
- For $f : X' \to X$ and $g : Y \to Y'$, $f \multimap g$ is a function sending a map $h : X \to Y$ to a map $g \circ h \circ f : X' \to Y'$. The realizer for $f \multimap g$ is $((x \mapsto \mathsf{r}_g \overleftarrow{@} ((\mathsf{r}_f \overleftarrow{@} x) \vec{@} w)) \leftarrow w)$.
- The evaluation map $ev : X \otimes (X \multimap Y) \to Y$ sends $x \otimes f$ to $f(x)$. The realizer is $(((x \vec{@} v \leftarrow v) \leftarrow x) \vec{@} w \leftarrow w)$.
- For any map $f : X \otimes Z \to Y$, there exists a unique map $g : Z \to X \multimap Y$ which satisfies $ev \circ (id_X \otimes g) = f$. This g is given as a function sending z to a function $x \mapsto f(x \otimes z)$. The realizer of g is $((x \mapsto \mathsf{r}_f \overleftarrow{@} (t \mapsto t \overleftarrow{@} x \overleftarrow{@} z)) \leftarrow z)$.
- For objects X and Y, the underlying set of $Y \multimap X$ is the set of maps from X to Y. $\|f\| := \{\mathsf{r} \mid \mathsf{r}$ is a realizer of $f\}$. ◀

Here what important is the way to take realizers of tensor products. We take the realizers as $\|x \otimes y\| := \{(z \mapsto z \overleftarrow{@} \mathsf{a} \overleftarrow{@} \mathsf{a}') \mid \mathsf{a} \in \|x\|_X, \mathsf{a}' \in \|y\|_Y\}$, while we would take $(z \overleftarrow{@} \mathsf{a} \overleftarrow{@} \mathsf{a}' \leftarrow z)$ if we define in the same way as Proposition 8 and Proposition 12.

▶ Remark 36. Take an object $A := (|\mathcal{A}|, \|\text{-}\|)$, where $\|\mathsf{a}\| := \{\mathsf{a}\}$. If we assume $\mathbf{Asm}(\mathcal{A})$ is an SMCC and the natural transformation for the symmetry sends $x \otimes y$ to $y \otimes x$, then there is a realizer r for the symmetry $A \otimes A \to A \otimes A$, which satisfies $\mathsf{r} \overleftarrow{@} (z \mapsto z \overleftarrow{@} \mathsf{a} \overleftarrow{@} \mathsf{a}') = (z \mapsto z \overleftarrow{@} \mathsf{a}' \overleftarrow{@} \mathsf{a})$ for arbitrary $\mathsf{a}, \mathsf{a}' \in |\mathcal{A}|$. Then we have $\overleftarrow{\mathsf{C}}' := ((\overleftarrow{\mathsf{I}} \vec{@} (\mathsf{r} \overleftarrow{@} (z \mapsto z \overleftarrow{@} x \overleftarrow{@} y)) \leftarrow y) \leftarrow x)$, which make $\mathcal{A}$ a BCI-algebra. Hence, when $\mathcal{A}$ is a bi-BDI-algebra and not a BCI-algebra, $\mathbf{Asm}(\mathcal{A})$ is not an SMCC (as long as we try to take the symmetry map in the natural way).

In the above proposition, we choose $\overleftarrow{@}$ to give $\mathbf{Asm}(\mathcal{A})$. However, it does not matter even if we choose $\vec{@}$.

▶ **Proposition 37.** *Let* $\mathcal{A} = (|\mathcal{A}|, \overleftarrow{@}, \vec{@})$ *be a bi-*BDI*-algebra. If we take an applicative structures* $\mathcal{A}' := (|\mathcal{A}|, \overleftarrow{@}')$ *for* $x \overleftarrow{@}' y := y \vec{@} x$*, then* $\mathbf{Asm}(\mathcal{A})$ *and* $\mathbf{Asm}(\mathcal{A}')$ *are isomorphic as categories. Moreover,* $\mathbf{Asm}(\mathcal{A})$ *is monoidally isomorphic to* $\mathbf{Asm}(\mathcal{A}')$ *with the reversed tensor products.*

We can also show $\mathbf{Mod}(\mathcal{A})$ is a monoidal bi-closed category for a bi-BDI-algebra $\mathcal{A}$. However, we cannot take tensor product as $\mathbf{Asm}(\mathcal{A})$ since $X \otimes Y$ in $\mathbf{Asm}(\mathcal{A})$ may not be a modest set even if X and Y are modest. We use a functor T that is the left adjoint of the inclusion functor $i : \mathbf{Mod}(\mathcal{A}) \hookrightarrow \mathbf{Asm}(\mathcal{A})$. T sends an assembly $(|X|, \|\text{-}\|_X)$ to a modest set $(|Z|, \|\text{-}\|_Z)$. Here $|Z| := |X|/\approx$, where the relation $\approx$ is the transitive closure of $\sim$ defined as $x \sim x' :\Leftrightarrow \|x\|_X \cap \|x'\|_X \neq \emptyset$. The realizers of $|Z|$ are $\|z\|_Z := \bigcup_{x \in z} \|x\|_X$. T sends a map f of $\mathbf{Asm}(\mathcal{A})$ to the canonical one of $\mathbf{Mod}(\mathcal{A})$, whose realizers are those of f. We define the tensor product $\boxtimes$ in $\mathbf{Mod}(\mathcal{A})$ as $X \boxtimes Y := T(iX \otimes iY)$. By using the same realizers in the proof of Proposition 35, we can prove that this $\boxtimes$ makes $\mathbf{Mod}(\mathcal{A})$ a monoidal bi-closed category. The same discussion for BCI-algebras is in [8] and the more general discussion about monoidal structures of reflective subcategories (for symmetric cases) is in [3].

▶ **Proposition 38.** $\mathbf{Mod}(\mathcal{A})$ *is a monoidal bi-closed category for a bi-*BDI*-algebra* $\mathcal{A}$*.*

To end of this subsection, we give a property about morphisms between bi-BDI-algebras.

▶ **Proposition 39.** *For bi-*BDI*-algebras* $(|\mathcal{A}_1|, \overleftarrow{@}_1, \vec{@}_1)$ *and* $(|\mathcal{A}_2|, \overleftarrow{@}_2, \vec{@}_2)$ *and an applicative morphism* $\gamma : (|\mathcal{A}_1|, \overleftarrow{@}_1) \to (|\mathcal{A}_2|, \overleftarrow{@}_2)$*,* $\gamma_* : \mathbf{Asm}(\mathcal{A}_1) \to \mathbf{Asm}(\mathcal{A}_2)$ *is a lax monoidal functor.*

▶ Remark 40. γ_* is not generally oplax monoidal since neither realizers for $\gamma_* I_1 \to I_2$ nor $\gamma_*(X \otimes_1 Y) \to \gamma_*(X) \otimes_2 \gamma_*(Y)$ exist.

4 Realizability models for modalities

In this section we relate our non-symmetric categorical realizability to the standard realizability based on BCI-algebras and PCAs. Our approach is similar to the case of linear combinatory algebras (LCAs) which relate BCI-algebras and PCAs. An LCA consists of a BCI-algebra $\mathcal{A}$, an endofunction $! : |\mathcal{A}| \to |\mathcal{A}|$ and several kinds of elements, such that the functor $!_*$ on $\mathbf{Asm}(\mathcal{A})$ (or $\mathbf{Mod}(\mathcal{A})$) becomes a linear exponential comonad on the SMCC [1]. While ! in an LCA is a function, in [8], ! is generalized to a total relation and the generalized LCAs are called relational linear combinatory algebras (rLCAs). We can obtain an rLCA from an adjoint pair between a BCI-algebra and a PCA.

The same construction can be applied to bi-BDI-algebras. That is, we can reformulate rLCAs for bi-BDI-algebras (Here we call *exp-rPLCAs*), and adjoint pairs between bi-BDI-algebras and PCAs induce exp-rPLCAs. Using exp-rPLCAs, we get models of !-modalities on non-symmetric multiplicative intuitionistic linear logic (MILL).

Also, we can obtain models for *exchange modalities* relating non-symmetric linear logics and symmetric MILL. A model of the logic with the exchange modality is given as a monoidal adjunction between an SMCC and a monoidal bi-closed category [9]. We can construct the model from a comonadic applicative morphism with certain conditions (Here we call an *exch-rPLCA*), and an adjoint pair between a bi-BDI-algebra and a BCI-algebra induces an exch-rPLCA.

4.1 Realizability models for !-modalities on non-symmetric linear logics

Linear exponential comonads on non-symmetric monoidal categories are investigated in [7], which model !-modalities on non-symmetric MILL.

▶ **Definition 41.** *A* linear exponential comonad ! *on a (non-symmetric) monoidal category* $\mathbf{C}$ *is a monoidal comonad on* $\mathbf{C}$ *such that the induced monoidal structure of the category of Eilenberg-Moore coalgebras is Cartesian.*

The above characterization is by Theorem 5 of [7]. However, originally linear exponential comonads are defined explicitly in [7] as tuples which consists of a monoidal comonad ! on $\mathbf{C}$ and monoidal natural transformations $e_X : !X \to I$ and $d_X : !X \to !X \otimes !X$ satisfying several conditions.

In this subsection, we generalize rLCAs to the non-symmetric case and show they give rise to linear exponential comonads.

▶ **Definition 42.** *An* exponential relational planar linear combinatory algebra *(exp-rPLCA) consists of a bi-*BDI*-algebra* $\mathcal{A} = (|\mathcal{A}|, \overleftarrow{\mathbb{D}}, \vec{\mathbb{a}})$ *and a comonadic applicative morphism* $(!, \mathsf{e}, \mathsf{d})$ *on* $(|\mathcal{A}|, \overleftarrow{\mathbb{D}})$ *which satisfies the following.*

- *There is* $\mathsf{k} \in |\mathcal{A}|$ *such that* $\mathsf{k} \overleftarrow{\mathbb{D}} \mathsf{x} \overleftarrow{\mathbb{D}} (!\mathsf{y}) \subseteq \{\mathsf{x}\}$ *for any* $\mathsf{x}, \mathsf{y} \in |\mathcal{A}|$.
- *There is* $\mathsf{w} \in |\mathcal{A}|$ *such that* $\mathsf{w} \overleftarrow{\mathbb{D}} \mathsf{x} \overleftarrow{\mathbb{D}} (!\mathsf{y}) \subseteq \mathsf{x} \overleftarrow{\mathbb{D}} (!\mathsf{y}) \overleftarrow{\mathbb{D}} (!\mathsf{y})$ *for any* $\mathsf{x}, \mathsf{y} \in |\mathcal{A}|$.

▶ **Proposition 43.** *For an exp-rPLCA* $(\mathcal{A}, !)$, $!_*$ *is a linear exponential comonad on* $\mathbf{Asm}(\mathcal{A})$.

This proposition is proven by giving realizers for natural transformations associated with a linear exponential comonad. $e_X : !_*X \to I$ sending x to $*$ has a realizer $\mathsf{k} \overleftarrow{\mathbb{D}} \overleftarrow{\mathsf{I}}$. $d_X : !_*X \to !_*X \otimes !_*X$ sending x to $x \otimes x$ has a realizer $\mathsf{w} \overleftarrow{\mathbb{D}} (((z \mapsto z \overleftarrow{\mathbb{D}} x \overleftarrow{\mathbb{D}} y) \leftarrow\!\shortmid y) \leftarrow\!\shortmid x)$.

Although now we get a linear exponential comonad $!_*$ on $\mathbf{Asm}(\mathcal{A})$, at this point it has not been concluded that we get linear-non-linear models (i.e., monoidal adjunctions between monoidal closed categories and CCCs) by categorical realizability since we have not shown that the co-Kleisli adjunction between $\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Asm}(\mathcal{A})_{!_*}$ is monoidal. In order to show that the co-Kleisli adjunction is indeed monoidal, it is enough to show that $\mathbf{Asm}(\mathcal{A})$ has Cartesian products. (It follows from Proposition 3 in [7].)

▶ **Proposition 44.** *For an exp-rPLCA* $(\mathcal{A}, !)$, $\mathbf{Asm}(\mathcal{A})$ *has Cartesian products, and thus the co-Kleisli adjunction between* $\mathbf{Asm}(\mathcal{A})$ *and a CCC* $\mathbf{Asm}(\mathcal{A})_{!_*}$ *is monoidal.*

The proof is almost the same as the proof of Proposition 12 of [8]. For instance, we take $\|(x, y)\| := \{(z \mapsto z \overleftarrow{\mathbb{D}} (w \mapsto w \overleftarrow{\mathbb{D}} u \overleftarrow{\mathbb{D}} v) \overleftarrow{\mathbb{D}} a) \mid \exists p, \exists q,\ u \in !p,\ v \in !q,\ p \overleftarrow{\mathbb{D}} a \in \|x\|_X$ and $q \overleftarrow{\mathbb{D}} a \in \|y\|_Y\}$ as realizers for Cartesian product. Here note that the Cartesian product $X \times Y$ is a modest set when X and Y are modest. Hence the above proposition for $\mathbf{Asm}(\mathcal{A})$ can be shown similarly for $\mathbf{Mod}(\mathcal{A})$. (See Remark 23 for restricting $!_*$ to $\mathbf{Mod}(\mathcal{A})$.)

▶ **Proposition 45.** *For an exp-rPLCA* $(\mathcal{A}, !)$, $!_*$ *is a linear exponential comonad on* $\mathbf{Mod}(\mathcal{A})$. *Furthermore,* $\mathbf{Mod}(\mathcal{A})$ *has Cartesian products and thus the co-Kleisli adjunction between* $\mathbf{Mod}(\mathcal{A})$ *and* $\mathbf{Mod}(\mathcal{A})_{!_*}$ *is monoidal.*

Next, we show that an adjoint pair between a bi-BDI-algebra and a PCA gives rise to an exp-rPLCA. First, note that it does not matter which application we choose when we take an adjoint pair, as shown in the next proposition.

▶ **Proposition 46.** *Let* $\mathcal{A} = (|\mathcal{A}|, \overleftarrow{\mathbb{D}}, \vec{\mathbb{a}})$ *be a bi-*BDI*-algebra and* $\mathcal{B} = (|\mathcal{B}|, \cdot)$ *be a* BCI*-algebra. Given an adjoint pair* $(\delta \dashv \gamma) : (|\mathcal{A}|, \overleftarrow{\mathbb{D}}) \to \mathcal{B}$, *there is an adjoint pair* $(\delta' \dashv \gamma') : (|\mathcal{A}|, \overleftarrow{\mathbb{D}'}) \to \mathcal{B}$ *for* $x \overleftarrow{\mathbb{D}'} y := y \vec{\mathbb{a}} x$, *such that* $\gamma = \gamma'$ *and* $\delta = \delta'$ *as total relations.*

▶ **Proposition 47.** *Let* $\mathcal{A} = (|\mathcal{A}|, \overleftarrow{\mathbb{D}}, \vec{\mathbb{a}})$ *be a bi-*BDI*-algebra and* $\mathcal{B} = (|\mathcal{B}|, \cdot)$ *be a PCA. For an adjoint pair* $(\delta \dashv \gamma) : \mathcal{A} \to \mathcal{B}$, $(\delta \circ \gamma)$ *forms an exp-rPLCA.*

Proof. Let e and d be realizers associated with $\delta \circ \gamma$ being a comonadic applicative morphism. $\mathsf{k} \in ((x \vec{@} (\mathsf{e} \overleftarrow{@} (\mathsf{r}_\delta \overleftarrow{@} \delta M \overleftarrow{@} y)) \leftarrow\!\shortmid y) \leftarrow\!\shortmid x)$, where $M \in \lambda^* z.(\gamma \vec{\mathsf{I}})$. $\mathsf{w} \in ((x \vec{@} (\mathsf{e} \overleftarrow{@} (\mathsf{r}_\delta \overleftarrow{@} \delta M \overleftarrow{@} (\mathsf{d} \overleftarrow{@} y))) \leftarrow\!\shortmid y) \leftarrow\!\shortmid x)$, where $M := \lambda^* z.\mathsf{r}_\gamma \cdot (\mathsf{r}_\gamma \cdot \gamma N \cdot z) \cdot z$ and $N := (((t \mapsto\!\shortmid t \overleftarrow{@} u \overleftarrow{@} v) \leftarrow\!\shortmid v) \leftarrow\!\shortmid u)$. ◀

Next, we give an example of an adjoint pair between a bi-BDI-algebra and a PCA. This example is a planar variant of the untyped linear lambda calculus with ! [13].

▶ **Example 48.** Suppose infinite supply of variables $x, y, z, \ldots$. Terms are defined grammatically as follows:

$$M ::= x \mid M \overleftarrow{@} M' \mid M \vec{@} M' \mid (M \leftarrow\!\shortmid x) \mid (x \mapsto\!\shortmid M) \mid !M \mid \lambda !x.M$$

Here x of $(M \leftarrow\!\shortmid x)$ (or $(x \mapsto\!\shortmid M)$) is the rightmost (or leftmost) free variable of M, appears exactly once in M and not in any scope of !. Take an equational relation on the terms as the congruence of the three equational axioms $(M \leftarrow\!\shortmid x) \overleftarrow{@} N = M[N/x]$, $N \vec{@} (x \mapsto\!\shortmid M) = M[N/x]$ and $(\lambda !x.M) \overleftarrow{@} (!N) = M[N/x]$. Let Λ be a set of equivalence classes of closed terms.

Then we obtain a bi-BDI-algebra $\mathcal{A} := (\Lambda, \overleftarrow{@}, \vec{@})$ and a PCA $\mathcal{B} := (\Lambda, \cdot)$, where $M \cdot N := M \overleftarrow{@} !N$. Here K and S exist in $\mathcal{B}$ as $\lambda !x.\lambda !y.x$ and $\lambda !x.\lambda !y.\lambda !z.x \overleftarrow{@} !z \overleftarrow{@} !(y \overleftarrow{@} !z)$.

Take an applicative morphism $\gamma : \mathcal{A} \to \mathcal{B}$ as the identity whose realizer is $\lambda !x.\lambda !y.x \overleftarrow{@} y$. Take $\delta : \mathcal{B} \to \mathcal{A}$ sending M to $!M$ whose realizer is $\lambda !x.\lambda !y.!(x \overleftarrow{@} !y)$. Then $\delta \dashv \gamma$.

4.2 Realizability models for exchange modalities

The Lambek calculus with the exchange modality and its categorical models are introduced in [9]. Here by "Lambek calculus" we mean the non-symmetric MILL with left and right implications. Its extension with the exchange modality is the *commutative/non-commutative (CNC) logic*, which is a sequent calculus composed of two (commutative and non-commutative) logics. Categorical models of CNC logics are given as monoidal adjunctions between monoidal bi-closed categories and SMCCs, and are called *Lambek adjoint models*. As well as exp-rPLCAs, we can define comonadic applicative morphisms giving rise to Lambek adjoint models.

▶ **Definition 49.** *An* exchange relational planar linear combinatory algebra *(exch-rPLCA) consists of a bi-*BDI*-algebra* $\mathcal{A} = (|\mathcal{A}|, \overleftarrow{@}, \vec{@})$ *and a comonadic applicative morphism* $(\xi, \mathsf{e}, \mathsf{d})$ *on* $\mathcal{A}$ *with* $\overleftarrow{\mathsf{c}} \in |\mathcal{A}|$ *satisfying* $\overleftarrow{\mathsf{c}} \overleftarrow{@} \mathsf{x} \overleftarrow{@} (\xi \mathsf{y}) \overleftarrow{@} (\xi \mathsf{z}) \subseteq \mathsf{x} \overleftarrow{@} (\xi \mathsf{z}) \overleftarrow{@} (\xi \mathsf{y})$ *for any* $\mathsf{x}, \mathsf{y}, \mathsf{z} \in |\mathcal{A}|$.

▶ **Proposition 50.** *For an exch-rPLCA* $(\mathcal{A}, \xi)$*, the co-Kleisli category* $\mathbf{Asm}(\mathcal{A})_{\xi_*}$ *is an SMCC and the co-Kleisli adjunction is monoidal.*

Proof (Sketch).

- We define tensor products in $\mathbf{Asm}(\mathcal{A})_{\xi_*}$ as $X \otimes' Y := (|X| \times |Y|, \|\text{-}\|)$, where $\|x \otimes' y\| := \{(z \mapsto\!\shortmid z \overleftarrow{@} \mathsf{a} \overleftarrow{@} \mathsf{a}') \mid \mathsf{a} \in \xi \|x\|_X \text{ and } \mathsf{a}' \in \xi \|y\|_Y\}$.
- For $f : X \to X'$ and $g : Y \to Y'$ in $\mathbf{Asm}(\mathcal{A})_{\xi_*}$, $f \otimes' g$ sends (x, y) to $(f(x), g(y))$. The realizer for $f \otimes' g$ is $(M \vec{@} (\mathsf{e} \overleftarrow{@} w) \leftarrow\!\shortmid w)$, where $M \in (((z \mapsto\!\shortmid z \overleftarrow{@} (\mathsf{r}_\xi \overleftarrow{@} (\xi \mathsf{r}_f) \overleftarrow{@} (\mathsf{d} \overleftarrow{@} x)) \overleftarrow{@} (\mathsf{r}_\xi \overleftarrow{@} (\xi \mathsf{r}_g) \overleftarrow{@} (\mathsf{d} \overleftarrow{@} y))) \leftarrow\!\shortmid y) \leftarrow\!\shortmid x)$.
- We define the unit object J in $\mathbf{Asm}(\mathcal{A})_{\xi_*}$ as $(\{*\}, \|\text{-}\|)$, where $\|*\| := \{\overleftarrow{\mathsf{I}}\}$.
- The symmetry $\sigma : X \otimes' Y \to Y \otimes' X$ sends $x \otimes' y$ to $y \otimes' x$. The realizers for σ and σ^{-1} are $(M \vec{@} (\mathsf{e} \overleftarrow{@} w) \leftarrow\!\shortmid w)$, where $M := \overleftarrow{\mathsf{c}} \overleftarrow{@} (((z \mapsto\!\shortmid z \overleftarrow{@} y \overleftarrow{@} x) \leftarrow\!\shortmid x) \leftarrow\!\shortmid y)$.
- For objects X and Y, the underlying set of the linear exponent $Y \multimap X$ is $Hom_{\mathbf{Asm}(\mathcal{A})}(\xi_* X, Y)$. $\|f\| := \{\mathsf{r} \in |\mathcal{A}| \mid \mathsf{r} \text{ is the realizer of } f\}$.

- For $f : X' \to X$ and $g : Y \to Y'$ in $\mathbf{Asm}(\mathcal{A})_{\xi_*}$, $g \circ\!\!- f$ sends a map $h : \xi_* X \to Y$ in $\mathbf{Asm}(\mathcal{A})$ to a map $g \circ (\xi_* h) \circ d_X \circ (\xi_* f) \circ d_{X'}$ from $\xi_* X'$ to Y' in $\mathbf{Asm}(\mathcal{A})$ (that is, a map from X' to Y' in $\mathbf{Asm}(\mathcal{A})_{\xi_*}$), where $d_X : \xi_* X \to \xi_* \xi_* X$ is the comultiplication of ξ_*. The realizer for $g \circ\!\!- f$ is $((\mathsf{r}_g \mathbin{\overleftarrow{@}} (\mathsf{r}_\xi \mathbin{\overleftarrow{@}} w \mathbin{\overleftarrow{@}} (\mathsf{d} \mathbin{\overleftarrow{@}} (\mathsf{r}_\xi \mathbin{\overleftarrow{@}} (\xi \mathsf{r}_f) \mathbin{\overleftarrow{@}} (\mathsf{d} \mathbin{\overleftarrow{@}} v)))) \leftarrow\!\!\mapsto v) \leftarrow\!\!\mapsto w)$.
- For any map $f : Z \otimes' X \to Y$ in $\mathbf{Asm}(\mathcal{A})_{\xi_*}$, there exists a unique map $g : Z \to Y \circ\!\!- X$ in $\mathbf{Asm}(\mathcal{A})_{\xi_*}$, which sends z to $x \mapsto f(z \otimes x)$. The realizer of g is $((\mathsf{r}_f \mathbin{\overleftarrow{@}} (\mathsf{r}_\xi \mathbin{\overleftarrow{@}} (\mathsf{r}_\xi \mathbin{\overleftarrow{@}} (\xi M) \mathbin{\overleftarrow{@}} (\mathsf{d} \mathbin{\overleftarrow{@}} z)) \mathbin{\overleftarrow{@}} (\mathsf{d} \mathbin{\overleftarrow{@}} x)) \leftarrow\!\!\mapsto x) \leftarrow\!\!\mapsto z)$, where $M := (((w \mapsto w \mathbin{\overleftarrow{@}} z \mathbin{\overleftarrow{@}} x) \leftarrow\!\!\mapsto x) \leftarrow\!\!\mapsto z)$.
- The co-Kleisli functor $\xi_* : \mathbf{Asm}(\mathcal{A})_{\xi_*} \to \mathbf{Asm}(\mathcal{A})$ is strong monoidal. Realizers for $\xi_* J \to I$ and $\xi_*(X \otimes' Y) \to \xi_* X \otimes \xi_* Y$ in $\mathbf{Asm}(\mathcal{A})$ are e. A realizer for $I \to \xi_* J$ is in $(w \mathbin{\overleftarrow{@}} (\xi \overleftarrow{\mathsf{I}}) \leftarrow\!\!\mapsto w)$. A realizer for $\xi_* X \otimes \xi_* Y \to \xi_*(X \otimes' Y)$ is in $(((\mathsf{r}_\xi \mathbin{\overleftarrow{@}} (\mathsf{r}_\xi \mathbin{\overleftarrow{@}} \xi M \mathbin{\overleftarrow{@}} (\mathsf{d} \mathbin{\overleftarrow{@}} x)) \mathbin{\overleftarrow{@}} (\mathsf{d} \mathbin{\overleftarrow{@}} y) \leftarrow\!\!\mapsto y) \leftarrow\!\!\mapsto x) \mathbin{\overrightarrow{@}} w \leftarrow\!\!\mapsto w)$, where $M := (((z \mapsto z \mathbin{\overleftarrow{@}} x \mathbin{\overleftarrow{@}} y) \leftarrow\!\!\mapsto y) \leftarrow\!\!\mapsto x)$. ◀

When we consider ξ_* on $\mathbf{Mod}(\mathcal{A})$, we can not use the same definition of the tensor product $\otimes'$ as $\mathbf{Asm}(\mathcal{A})_{\xi_*}$ since $X \otimes' Y$ may not be a modest set. We again use the functor T that is the left adjoint of the inclusion functor $i : \mathbf{Mod}(\mathcal{A}) \hookrightarrow \mathbf{Asm}(\mathcal{A})$ and define the tensor product $X \boxtimes Y$ in $\mathbf{Mod}(\mathcal{A})_{\xi_*}$ as $T(iX \otimes' iY)$. Then we can prove that $\mathbf{Mod}(\mathcal{A})_{\xi_*}$ becomes an SMCC with $\boxtimes$ in the same way as Proposition 50.

▶ **Proposition 51.** *For an exch-rPLCA $(\mathcal{A}, \xi)$, the co-Kleisli category $\mathbf{Mod}(\mathcal{A})_{\xi_*}$ is an SMCC and the co-Kleisli adjunction is monoidal.*

Similar to exp-rPLCAs, we can obtain an exch-rPLCA from an adjoint pair.

▶ **Proposition 52.** *Let $\mathcal{A} = (|\mathcal{A}|, \mathbin{\overleftarrow{@}}, \mathbin{\overrightarrow{@}})$ be a bi-*BDI*-algebra and $\mathcal{B} = (|\mathcal{B}|, \cdot)$ be a* BCI*-algebra. For an adjoint pair $(\delta \dashv \gamma) : \mathcal{A} \to \mathcal{B}$, $(\delta \circ \gamma)$ forms an exch-rPLCA.*

Proof. Let e and d be realizers associated with $\delta \circ \gamma$ being a comonadic applicative morphism. $\overleftarrow{\mathsf{c}} \in (((x \mathbin{\overrightarrow{@}} (\mathsf{e} \mathbin{\overleftarrow{@}} (\mathsf{r}_\delta \mathbin{\overleftarrow{@}} (\mathsf{r}_\delta \mathbin{\overleftarrow{@}} \delta M \mathbin{\overleftarrow{@}} (\mathsf{d} \mathbin{\overleftarrow{@}} y)) \mathbin{\overleftarrow{@}} (\mathsf{d} \mathbin{\overleftarrow{@}} z))) \leftarrow\!\!\mapsto z) \leftarrow\!\!\mapsto y) \leftarrow\!\!\mapsto x)$, where $M \in \lambda^* y.\lambda^* z.\mathsf{r}_\gamma \cdot (\mathsf{r}_\gamma \cdot \gamma N \cdot z) \cdot y$ and $N := (((x \mapsto x \mathbin{\overleftarrow{@}} z \mathbin{\overleftarrow{@}} y) \leftarrow\!\!\mapsto y) \leftarrow\!\!\mapsto z)$. ◀

▶ **Example 53.** Take an ordered group $(G, \cdot, e, \leq)$. Let T be the same set as in Example 32. We get a BCI-algebra $\mathcal{T}'$ as the powerset of $\{t \in T \mid e = |t|\}$, where the application is the same as the right application of $\mathcal{T}$ and $\mathsf{C} := \{((t_1 \circ\!\!- t_2) \circ\!\!- t_3) \circ\!\!- ((t_1 \circ\!\!- t_3) \circ\!\!- t_2) \mid |t_i| = e\}$.

In Example 32, we get a bi-BDI-algebra $\mathcal{T}$. We obtain two applicative morphisms $\gamma : \mathcal{T} \to \mathcal{T}'$ as a function $M \mapsto \{(t \circ\!\!- t) \mid t \in M\}$ and $\delta : \mathcal{T}' \to \mathcal{T}$ as the identity function. Here the realizer for γ is $\{((t_1 \circ\!\!- t_1) \circ\!\!- (t_2 \circ\!\!- t_2)) \circ\!\!- ((t_1 \circ\!\!- t_2) \circ\!\!- (t_1 \circ\!\!- t_2)) \mid t_1, t_2 \in T\}$ and the realizer for δ is $\{t \circ\!\!- t \mid t \in T\}$. γ and δ form an adjoint pair, where the realizer for $id \preceq \gamma \circ \delta$ is $\{t \circ\!\!- (t \circ\!\!- t) \mid e \leq |t|\}$ and the realizer for $\delta \circ \gamma \preceq id$ is $\{((t \circ\!\!- t) \circ\!\!- (t \circ\!\!- t)) \circ\!\!- (t \circ\!\!- t) \mid t \in T\}$.

▶ Remark 54. The above construction can not be applied to exp-rPLCAs. No matter how we take a powerset $\mathcal{T}_0$, since $M \mathbin{\overleftarrow{@}} N \mathbin{\overleftarrow{@}} \emptyset = \emptyset$ for any $M, N \in \mathcal{T}_0$, $\mathcal{T}_0$ does not contain the element acting as the K-combinator and thus cannot be a PCA.

5 Related work

In [15], the relationships between the planar lambda calculus and planar maps are investigated. It is shown that we can generate rooted planar maps with orientations by combining a few kinds of "imploid moves", that corresponds to the combinatory completeness of $\mathsf{BI}(-)^\bullet$-algebras and the planar lambda calculus. We may apply the correspondence between planar

lambda terms and rooted planar maps to our bi-planar lambda terms. Here the corresponding rooted maps have two kinds of vertexes with two inputs and one output (or, two outputs and one input) while rooted maps for planar lambda terms have one kind.

Although we use the word "Lambek calculus" as a variant of non-symmetric MILL with left and right implications in this paper, the word "Lambek calculus" has various meanings as logics. The basics about the Lambek calculus is in [10]. Our treatment in this paper is from [9].

Conditions of bi-BDI-algebras may look like "dual combinators" introduced in [4]. In both of them, elements acting as functions can act to an argument from both left and right sides. However, a dual combinatory logic has only one sort of application and the reductions does not satisfy the confluence, whereas bi-BDI-algebras have two sorts of applications and the confluence for the bi-planar lambda calculus holds.

Realizability for (symmetric) linear exponentials are introduced in [1] as LCAs and generalized to rLCAs in [8]. Section 4.1 of this paper is the reformulations of some contents of [8] to the planar case. The original definition of rLCA is described using not k and w but $\preceq$. We can also define exp-rPLCAs without using k nor w, however, we are not sure whether we can define exch-rPLCAs without using $\overline{\mathsf{c}}$.

6 Conclusion

In this paper, we presented a new class of applicative structures called bi-BDI-algebras. Bi-BDI-algebras lie between $\mathsf{BI}(-)^\bullet$-algebras and BCI-algebras and correspond to the bi-planar lambda calculus. Given a bi-BDI-algebra $\mathcal{A}$, we obtain monoidal bi-closed categories $\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Mod}(\mathcal{A})$. We also introduced exp-rPLCAs and exch-rPLCAs which induce categorical models for !-modalities and exchange modalities on non-symmetric logics. We can get exp-rPLCAs from adjoint pairs between bi-BDI-algebras and PCAs, and exch-rPLCAs from adjoint pairs between bi-BDI-algebras and BCI-algebras.

We conclude this paper by describing three issues for future work. First, while we have shown that a bi-BDI-algebra $\mathcal{A}$ induces a monoidal bi-closed category $\mathbf{Asm}(\mathcal{A})$, it is not clear whether being $\mathbf{Asm}(\mathcal{A})$ a monoidal bi-closed category leads that $\mathcal{A}$ is a bi-BDI-algebra. For some other classes, we can show such a proposition. For instance, being $\mathbf{Asm}(\mathcal{A})$ a CCC/SMCC/closed multicategory leads that $\mathcal{A}$ is a PCA/BCI-algebra/$\mathsf{BI}(-)^\bullet$-algebra under some natural conditions. (See Proposition 19 of [14] for the case of $\mathsf{BI}(-)^\bullet$-algebras.)

Second, there are a few points that exp-rPLCA and exch-rPLCA do not behave in the same way, and we would like to clarify them. As we said in the previous section, while exp-rPLCA can be defined without using k nor w, we are not sure that exch-rPLCA can be defined in such a style. Also, while an adjoint pair between a bi-BDI-algebra $\mathcal{A}$ and a PCA $\mathcal{B}$ induce a monoidal adjunction between $\mathbf{Asm}(\mathcal{A})$ and $\mathbf{Asm}(\mathcal{B})$, the adjunction between $\mathbf{Asm}(\mathcal{A}')$ and $\mathbf{Asm}(\mathcal{B}')$ induced from an adjoint pair between a bi-BDI-algebra $\mathcal{A}'$ and a BCI-algebra $\mathcal{B}'$ is not generally monoidal.

Finally, we are yet to find more interesting concrete examples of bi-BDI-algebras and adjoint pairs, which should be useful for investing non-commutative logics and their models in a systematic way.

References

1 Samson Abramsky, Esfandiar Haghverdi, and Philip Scott. Geometry of interaction and linear combinatory algebras. *Mathematical Structures in Computer Science*, 12(5):625–665, 2002.

2 Samson Abramsky and Marina Lenisa. Linear realizability and full completeness for typed lambda-calculi. *Annals of Pure and Applied Logic*, 134(2-3):122–168, 2005.

3 Brian Day. A reflection theorem for closed categories. *Journal of pure and applied algebra*, 2(1):1–11, 1972.

4 J Michael Dunn and Robert K Meyer. Combinators and structurally free logic. *Logic Journal of IGPL*, 5(4):505–537, 1997.

5 Jean-Yves Girard. Linear logic. *Theoretical computer science*, 50(1):1–101, 1987.

6 Masahito Hasegawa. A quantum double construction in Rel. *Mathematical Structures in Computer Science*, 22(4):618–650, 2012.

7 Masahito Hasegawa. Linear exponential comonads without symmetry. *Electronic Proceedings in Theoretical Computer Science*, 238:54–63, 2016.

8 Naohiko Hoshino. Linear realizability. In *International Workshop on Computer Science Logic*, pages 420–434. Springer, 2007.

9 Jiaming Jiang, Harley Eades III, and Valeria de Paiva. On the lambek calculus with an exchange modality. In Thomas Ehrhard, Maribel Fernández, Valeria de Paiva, and Lorenzo Tortora de Falco, editors, *Proceedings Joint International Workshop on Linearity & Trends in Linear Logic and Applications, Linearity-TLLA@FLoC 2018, Oxford, UK, 7-8 July 2018*, volume 292 of *EPTCS*, pages 43–89, 2018.

10 Joachim Lambek. Deductive systems and categories II. standard constructions and closed categories. In *Category theory, homology theory and their applications I*, pages 76–122. Springer, 1969.

11 John R Longley. *Realizability toposes and language semantics*. PhD thesis, University of Edinburgh, 1995.

12 Oleksandr Manzyuk. Closed categories vs. closed multicategories. *Theory and Applications of Categories*, 26(5):132–175, 2012.

13 Alex Simpson. Reduction in a linear lambda-calculus with applications to operational semantics. In *International Conference on Rewriting Techniques and Applications*, pages 219–234. Springer, 2005.

14 Haruka Tomita. Realizability Without Symmetry. In *29th EACSL Annual Conference on Computer Science Logic (CSL 2021)*, volume 183 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 38:1–38:16. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2021.

15 Noam Zeilberger. A theory of linear typings as flows on 3-valent graphs. In *Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 919–928. ACM, 2018.

16 Noam Zeilberger and Alain Giorgetti. A correspondence between rooted planar maps and normal planar lambda terms. *Log. Methods Comput. Sci.*, 11, 2015.