Logic and computation as combinatorics

Norihiro Yamada

yamad041@umn.edu
University of Minnesota

June 3, 2023

Abstract

The *syntactic* nature of logic and computation separates them from other fields of mathematics. Nevertheless, syntax has been the only way to adequately capture the *dynamics* of proofs and programs such as cut-elimination, and the *finiteness* and the *atomicity* of syntax are preferable for foundational aims as seen in Hilbert's program. Another issue is that a *uniform* basis for logic and computation has been missing, and this problem hampers a coherent view on them. For instance, formal proofs in proof theory are far from (ordinary) proofs of the validity of a formula in model theory. Our goal is to solve these fundamental problems by rebuilding central concepts in logic and computation such as formal systems, validity (in such a way that it coincides with the existence of proofs), cut-elimination and computability uniformly in terms of finite graphs based on *game semantics*. Unlike game semantics, however, we do not rely on anything infinite or extrinsic to the graphs. A key idea that enables our finitary, autonomous approach is the shift from graphs in game semantics to *dynamic* ones. The resulting combinatorics establishes a single, syntax-free, finitary framework that recasts formal systems admitting proofs with *cuts*, validity, the *finest* computational steps of cut-elimination and *higher-order* computability. This subsumes *fully complete* semantics of intuitionistic linear logic, which solves a problem open for thirty years, and even extends the full completeness to proofs with cuts. As a byproduct, our dynamic graphs give rise to *Hopf algebras*, which opens up new applications of algebras to logic and computation.

Contents

1 Introduction

Logic and computation have been suffering from the lack of a syntax-free or uniform foundation. We take a step towards a solution to this fundamental problem by reducing them to the study of finite dynamic graphs. The resulting work recasts formulae, proofs (not only provability) and computability *syntax-freely* and *uniformly*, going beyond the combinatorial characterisation of classical provability by Hughes [Hug06]. This result includes a complete solution to a bottleneck of the problem: only syntactically defined *dynamics* and *intensionality* in logic and computation. It also solves a problem open for thirty years: *full completeness* for intuitionistic linear logic.

1.1 Foreword

On the one hand, the *syntactic* nature of (formal) proofs and programs separates mathematical logic and theoretical computer science, or logic and computation for short, from other branches of mathematics. As cited by Hughes [Hug06], for instance, De Morgan [DM68] states that

[M]athematicians care no more for logic than logicians for mathematics.

The issue is that mathematics studies the *general, abstract essence* of the universe, but syntactic objects such as proofs and programs are bound by *how to write*, i.e., inessential details.

Besides, from a mathematician's view, syntax-independent objects such as sets and functions come first, and syntax is nothing but their notations. For this point of view, syntax *per se* is mysterious and cumbersome. For example, what proofs are and why they validate formulae are central questions in logic, but the standard approach of the field takes *how to write proofs* as the definition of proofs and justifies these notations as validations of formulae only in terms of how to rewrite proofs [Gen35, Pra71, Dum93] or indirectly by their completeness [Göd29]. This syntactic paradigm does not clarify that much what proofs are or why they validate formulae.

Strictly speaking, the field of *mathematical semantics* [Sco70, Gun92] has established a variety of syntax-free interpretations of the *extensional* aspects of proofs (though most of them do not look like proofs in the practice of mathematics) and programs. However, no syntax-free method has completely replaced proofs or programs because syntax has been the only way to adequately model their *dynamics* such as cut-elimination [Gen35]. This *intensional* concept is central in logic and computation yet hard to capture syntax-freely. Abramsky [Abr14] describes this problem as

Extensionality is enshrined in mathematically precise axioms with a clear conceptual meaning. Intensionality, by contrast, remains elusive. It is a "loose baggy monster" into which all manner of notions may be stuffed, and a compelling and coherent general framework for intensional concepts is still to emerge.

Last but not least, syntactic concepts are often defined *inductively*, lacking direct or intuitive descriptions. This inductive nature blocks a deeper analysis of logic or computation. For instance, consider classical logic [Fre79], intuitionistic logic [Hey30] and linear logic [Gir87]. They are very basic concepts in logic, but it is hard to compare or relate them in a direct or intuitive fashion because their proofs are defined only inductively.

In addition to the syntactic nature, another problem in logic and computation is the lack of a *common* framework, which makes it difficult to obtain a coherent view, a universal language or a shared mathematical technique on the fields. For instance:

- Formal proofs of a formula are not a formalisation of (ordinary) proofs of the validity of the formula, and they are only indirectly connected via soundness or completeness theorems. This illustrates a *schism* between proof theory and model theory.
- Set theory is based on not only sets but also (classical) logic. In other words, sets do not serve as a single principle of mathematics since otherwise they would not require logic.
- Recursion theory entails models of computations such as Turing machines [Tur37], which are *extrinsic* to the structure of sets or logic [Abr14, §1.2]. For instance, the computability of an arithmetic function depends on the existence of a Turing machine that computes the function, but the Turing machine itself is a priori extrinsic to the function.

This situation is in contrast with topology since general topology serves as a *single* foundation of the subfields of topology (on the basis of sets and logic). It is not only coherent but also elegant and significant if a single principle founds logic and computation, thus mathematics too. Such a principle will build new connections between their subfields and advance the fields as a whole.

Finally, logic and computation, being foundations of mathematics, ought to be *justifiable.* In particular, they should only rely on simple, primitive or *atomic* concepts. For instance, Hilbert's program [Hil31] was motivated by this standpoint, and its aim was to reduce all mathematics to *finitary* one. The point of such finitary mathematics, in Hilbert's own words [Hil26], is that

> If logical inference is to be reliable, it must be possible to survey these objects completely in all their parts, and the fact that they occur, that they differ from one another, and that they follow each other, or are concatenated, is immediately given intuitively, together with the objects, as something that can neither be reduced to anything else nor requires reduction.

Also, a justification of a framework for logic or computation should entail *conceptual naturality* as well, but we do not regard this point as primary because it can be subjective.

Motivated in this way, we propose:

Our research program. To reduce logic and computation to single, syntax-free, finitary mathematics, viz., *combinatorics*, that is conceptually natural for logic and computation.

The aim of the present work is to take a significant step for this research program:

The goal of this work. To reformulate formal systems for classical, intuitionistic and linear propositional logics, the validity of a formula, cut-elimination and computability uniformly and naturally by a class of finite dynamic graphs, called *combinatorial arenas.*

Remark. We leave it to another article to extend the present framework to other central concepts such as predicate logics, sets and computational complexity; see §1.3 for an outline.

Our framework has the following features. First, inherited from *game semantics* [Hyl97], a subfield of mathematical semantics, our approach is *conceptually natural.* Indeed, a combinatorial arena $\mathscr{A}$ represents a *game* between an agent, called *Player*, and an oracle, called *Opponent*, and a class of walks on $\mathscr{A}$ alternating between Player and Opponent, called *plays* on $\mathscr{A}$, depict their *dialogical arguments* or *computations.* A class of algorithms for Player about how to play on $\mathscr{A}$ are called *strategies* on $\mathscr{A}$, and a strategy is said to be *winning* if it always leads to Player's 'win.' We then interpret formulae and types by combinatorial arenas, and proofs (respectively, programs) by winning strategies (respectively, strategies). In this way, our combinatorics formalises the intuitive idea that reasoning and computing are certain kinds of playing games.

Besides, this approach resolves the schism between proof theory and model theory by defining a winning strategy to be not only a formal proof but also a *validation* of a formula. This definition makes sense because intuitively a winning strategy defends the validity of a formula against refutations by Opponent. That is, winning strategies can be seen, unlike other mathematical semantics, as an idealisation of proofs. In addition to this compelling intuition, technical virtues of the unification are that the consistency of our combinatorial formal systems follows immediately from a nature of winning strategies, and it is vacant to ask their soundness or completeness.

While our approach is based on game semantics, it has distinguished features:

- Our method is *combinatorial*, viz., it only uses finite, atomic structures intrinsic to combinatorial arenas, while game semantics is not (§1.3). This is desirable not only for foundational but also mathematical reasons: Combinatorial arenas form *Hopf algebras*, and it is *polynomial time decidable* whether a given strategy on a combinatorial arena is winning.
- We establish bijections between proofs, which may contain *cuts*, and winning strategies for classical, intuitionistic and linear logics, respectively, in a *non-inductive* fashion (§1.2).
- Game semantics does not formalise cut-elimination or computability, but our combinatorics does both, significantly improving *dynamic game semantics* [YA20, Yam19] (§1.2).

Remark. Although our combinatorial structures do not require anything infinite, we shall employ infinitary objects on the way for convenience but eventually dispense with them.

1.2 Main results

Concretely, we achieve the aim of §1.1 by proving the following theorems. To state the theorems, we first need the following definitions. There are constructions on combinatorial arenas, *negation* $\neg$, *linear implication* $\multimap$, *of-course* ! and *why-not* ?, coming from linear logic. Intuitively, $\neg\mathscr{A}$ is the negation of $\mathscr{A}$, $\mathscr{A} \multimap \mathscr{B}$ is the space of linear maps from $\mathscr{A}$ to $\mathscr{B}$, $!\mathscr{A}$ is the countable copies of $\mathscr{A}$, and $?\mathscr{A}$ is $\mathscr{A}$ itself except that strategies on $?\mathscr{A}$ can *backtrack* any number of times.

We then define a bicategory $\mathcal{LG}$, whose 0-cells are combinatorial arenas, 1-cells $\mathscr{A} \to \mathscr{B}$ are a class of finite winning strategies whose values under a combinatorial recast of cut-elimination are on $\mathscr{A} \multimap \mathscr{B}$, and 2-cells are the equivalence relation that identifies 1-cells up to the combinatorial cut-elimination. By this bicategorical framework, which was originally introduced in [YA20], $\mathcal{LG}$ admits intensionality or *cuts* in strategies. We also define a bicategory $\mathcal{LG}_{\neg\neg}$ (respectively, $\mathcal{LG}_{!}$, $\mathcal{LG}_{!?}$), whose 0-cells are a class of combinatorial arenas, 1-cells $\mathscr{A} \to \mathscr{B}$ are the same strategies whose values are on $\mathscr{A} \multimap \neg\neg\mathscr{B}$ (respectively, $!\mathscr{A} \multimap \mathscr{B}$, $!?\mathscr{A} \multimap ?\mathscr{B}$), and 2-cells are as those in $\mathcal{LG}$. Categorically, these bicategories are obtained from $\mathcal{LG}$ by (co-)Kleisli constructions.

Correspondingly, we define a syntactic bicategory ILL of intuitionistic linear logic, whose 0-cells are formulae, 1-cells $A \to B$ are (formal) proofs whose values under cut-elimination are on the sequent $A \vdash B$, and 2-cells are the equivalence relation that identifies 1-cells up to cut-elimination. In the same vein, we also define syntactic bicategories CLL, IL and CL of classical linear, intuitionistic and classical logics, respectively. We can now state our first theorem:

Theorem (combinatorial formal systems)**.** *There are biequivalences* $\mathsf{ILL} \simeq \mathcal{LG}$*,* $\mathsf{CLL} \simeq \mathcal{LG}_{\neg\neg}$*,* $\mathsf{IL} \simeq \mathcal{LG}_{!}$ *and* $\mathsf{CL} \simeq \mathcal{LG}_{!?}$*, and it is polynomial time decidable if a strategy is a 1-cell in all cases.*

This theorem characterises formulae and proofs of the logics syntax-freely by the same combinatorics. Also, for each case, it takes only polynomial time to check if a strategy is a 1-cell. Thus, our combinatorics constitutes *proof (formal) systems* in the sense of [CR79]. As a result, this theorem covers not only what is equivalent to Hughes' combinatorial reformulation of provability of classical logic [Hug06] but also that of intuitionistic and linear logics *uniformly*.

Whilst Hughes only characterises *provability* of classical logic, our biequivalences $[\![\_]\!]$ consist of bijections between strategies and *proofs*. Such a bijection is one of the strongest theorems in mathematical semantics, and it is often very difficult to establish. For instance, fully complete semantics of intuitionistic linear logic (with respect to cut-free proofs) was missing for thirty years (since the emergence of fully complete semantics of some fragments of linear logic [AJ94, HO93]), but the biequivalence $\mathsf{ILL} \simeq \mathcal{LG}$ solves this well-known problem. Besides, while full completeness has been focusing on cut-free proofs, the biequivalence admits proofs with *cuts*. Moreover, albeit semantics is usually given inductively, our biequivalences directly and non-inductively read off proofs as strategies, and vice versa. These progresses are made possible by the novel structure of combinatorial arenas. Lastly, although there is no known intuitive reading of Hughes' approach, ours has one: A combinatorial arenas is a game on the truth of a formula, and a winning strategy is an algorithm for Player to defend the truth of a formula against refutations by Opponent.

For this intuition, it makes sense to define a formula A to be *valid* if there is a 1-cell $1 \to [\![A]\!]$ in the four bicategories, where 1 is a terminal object that admits the isomorphism $[\![A]\!] \cong [\![A]\!]^1$. This definition makes the existence of a proof and the validity of a formula *coincide*.

Besides, our theorem characterises *non-linearity* and *classicality* of logic non-inductively by the (co-)Kleisli constructions. This classification has an intuitive reading too: *Non-linearity* of logic is obtained by the co-Kleisli construction $(\_)_!$, which enables strategies to *consume inputs any number of times*, and *classicality* by the Kleisli one $(\_)_?$, which permits strategies to *backtrack any number of times*. For example, the law of excluded middle, i.e., the disjunction $A \vee \neg A$ between each formula A and its negation $\neg A$, is provable in classical logic but not in intuitionistic logic. Our method explains this difference non-inductively and intuitively as follows. If the same symbol $\vee$ denotes the corresponding operation on combinatorial arenas, then the law means the existence of a 1-cell $1 \to [\![A]\!] \vee \neg[\![A]\!]$. There is no such 1-cell in $\mathcal{LG}_!$ as the choice between $[\![A]\!]$ and $\neg[\![A]\!]$ is not always decidable. In contrast, there *is* one in $\mathcal{LG}_{!?}$ since why-not ? on $[\![A]\!] \vee \neg[\![A]\!]$ allows the 1-cell to go back and forth between $[\![A]\!]$ and $\neg[\![A]\!]$ (without deciding which). That is, classical logic permits backtracks or *reasoning do-overs*, while intuitionistic logic does not.

Next, our second theorem follows immediately from the biequivalences that subsume cuts:

Theorem (combinatorial cut-elimination)**.** *The cut-elimination on 1-cells in $\mathcal{LG}$ (respectively, $\mathcal{LG}_{\neg\neg}$, $\mathcal{LG}_!$, $\mathcal{LG}_{!?}$) corresponds precisely to the one in* ILL *(respectively,* CLL*,* IL*,* CL*).*

To the best of our knowledge, this is the first syntax-free characterisation of cut-elimination, while there were partial solutions in the literature. For example, geometry of interaction [Gir89] models cut-elimination for the first time, but it deletes all (semantic) cuts *in one go*; dynamic game semantics [YA20] improves this pioneering work by modelling *step-by-step* reduction, but it is still much coarser than cut-elimination. In contrast, our second theorem completely captures cut-elimination, including its *finest* computational steps. Recall that the bottleneck in releasing logic from the syntactic occupation is dynamics and intensionality (§1.1). Our first and second theorems solve this problem (on propositional logics) entirely for the first time.

Further, our combinatorics, despite its finitary nature, forms a strong model of higher-order computation even without relying on any other models of computation. This is possible roughly because the combinatorics has intensional structures sufficient to define computation. Concretely, we obtain bicategories $\mathcal{G}$, $\mathcal{G}_{\neg\neg}$, $\mathcal{G}_!$ and $\mathcal{G}_{!?}$ respectively from $\mathcal{LG}$, $\mathcal{LG}_{\neg\neg}$, $\mathcal{LG}_!$ and $\mathcal{LG}_{!?}$ by removing the winning constraint on 1-cells and replacing their finiteness with *finite presentability* in a suitable sense. 1-cells in these bicategories model computations, more general than proofs, and in particular some of them are non-terminating or *partial*. Then, our last theorem is

Theorem (combinatorial computation)**.** *The bicategory $\mathcal{G}_!$ forms a model of computation that can simulate the higher-order functional programming language PCF [Sco93, Plo77].*

This theorem significantly improves Yamada [Yam19], which shows that strategies *presentable by finitely presentable strategies* can simulate PCF, as our theorem proves that finitely presentable strategies suffice. The mechanism behind this improvement is that our combinatorial approach frees strategies from the computation on infinitely many copies of games in [Yam19]. Another notable feature of this novel model of computation is its semantic or *abstract* nature: It is free from the symbolic computation of Turing machines or λ-calculi. This feature is quite desirable because it saves us from being bothered by inessential details in symbolic computation.

The bicategories $\mathcal{G}$, $\mathcal{G}_{\neg\neg}$ and $\mathcal{G}_{!?}$ model computation too, where $\mathcal{G}_{!?}$ interestingly combines computation and classical reasoning. We leave it as future work to analyse these bicategories.

Last but not least, our combinatorics gives rise to a well-known algebraic structure:

Proposition (combinatorial Hopf algebras)**.** *Combinatorial arenas constitute Hopf algebras.*

Over the past decades, Hopf algebras arising in combinatorics, or *combinatorial Hopf algebras* [JR79, ABS06], have been extensively studied. The present result uncovers a bridge between our combinatorics and this vibrant line of research. In addition to the value of connecting previously unrelated notions, this bridge enables one to apply methods and results in combinatorial Hopf algebras to logic and computation, e.g., for counting the number of formulae and proofs.

1.3 Our contributions and related work

Our first contribution is the *biequivalences* between formal systems and combinatorics. The main breakthrough here is that the biequivalences admit proofs with *cuts*, which in turn enables the combinatorics to model *cut-elimination*. In this way, we resolve the bottleneck in releasing logic from the syntactic occupation, i.e., dynamics and intensionality, completely for the first time (to the best of our knowledge). In a broader perspective, this result establishes a framework that tames the 'loose baggy monster' cited in §1.1 so that mathematics may extend its scope from static, extensional structures such as sets and functions to dynamic, intensional ones.

The significance of the biequivalences is visible even if one focuses on cut-free proofs: Fully complete semantics of intuitionistic linear logic with respect to cut-free proofs was open for thirty years, but a solution to this long-standing problem just follows from the cut-free part of the biequivalence $\mathsf{ILL} \simeq \mathcal{LG}$. This result 'beats the end boss' since even the fully complete semantics of classical linear logic [AM99c, Mel05] and of the multiplicative fragment of intuitionistic linear logic [MO03] was established about twenty years ago. Another implication of the biequivalences is that they provide a direct reading of our combinatorics as formal calculi equipped with cut-elimination. In this fashion, our method retains the *mechanical* nature of syntax.

Our second contribution is to extend the combinatorics for logic to computation. Thus, we have achieved our goal to reduce logic and computation *uniformly* to combinatorics (§1.1). The significance of the resulting model of computation is that it greatly improves the main result of Yamada [Yam19] by showing that finitely presentable strategies suffice to simulate PCF. Also, the computational steps of our model are more *explicit* and *atomic* than those of [Yam19], similarly to Turing machines yet in a *non-symbolic*, *higher-order* setting. Let us leave it as future work to extend this framework to a basis of *higher-order computational complexity*, whose mathematical foundation has not been established yet, by exploiting the explicit, atomic nature.

Our last contribution is the Hopf algebras induced by combinatorial arenas. This result uncovers a link between our combinatorics and a well-known algebraic structure. We add that combinatorial arenas also form Hausdorff topological spaces, and strategies continuous maps, while *domains*, the best-known structure in mathematical semantics, only give rise to non-Hausdorff topological spaces. Although we leave it as future work to explore these connections, they indicate fruitful interplays between our combinatorics and traditional branches of mathematics.

Game semantics has been extended to *Martin-Löf type theory (MLTT)* [AJV15, Yam19], which subsumes intuitionistic higher-order predicate logic, and Aczel [Acz86] has shown that constructive set theory is translatable into MLTT. We shall therefore extend the present framework to predicate logics and set theory through the game semantics of MLTT.

For related work, we have made comparisons with Hughes' combinatorial recast of provability of classical logic [Hug06], geometry of interaction [Gir89], dynamic game semantics [YA20] and the game-semantic model of computation [Yam19] (§1.2). In the following, we list related work in game semantics. A major approach pioneered by Hyland and Ong [HO00] or *HO* is to replace games with a class of finite rooted dags, called *arenas*, and then *derive* possible developments or *positions* in an arena as a class of walks on the arena. This approach does not require anything infinite, and our method is based on HO for this reason. However, HO cannot model linear logic, in particular copying ! nor backtracking ? which are crucial for this work, due to the lack of a structure to control positions. Besides, positions in an arena may grow *infinitely* even when they model finitary objects such as natural numbers, and this problem makes it impossible for HO to model a *termination* of a play. Our initial idea was to shift from arenas to combinatorial arenas, or from dags to dynamic simple graphs, so that it solves the aforementioned problems of HO.

Another major approach pioneered by Abramsky et al. [AJ94, AJM00] or *AJM* is to *assign* finite sequences to a game as positions in the game. This method is more abstract and general than HO or ours as it does not go through (combinatorial) arenas. Meanwhile, the AJM method entails *infinite* sets of positions. Also, while our combinatorics achieves semantics not only fully faithful but also *essentially surjective on objects*, it is unclear how the AJM approach does it.

Based on AJM, Murawski and Ong [MO03] achieved fully complete game semantics of the multiplicative fragment of intuitionistic linear logic. Laurent [Lau04, Lau05] constructed, on the basis of HO and AJM, fully complete game semantics of the *polarised* fragment of linear logic. His games, however, do not even give rise to a category. The fully complete semantics of classical linear logic [AM99c, Mel05] uses *concurrent* games, while our and the aforementioned semantics are based on *sequential* games. Similarly to AJM, concurrent games entail infinitary structures.

1.4 Structure of the present article

We first introduce combinatorial arenas and finitely presentable strategies in §2. We next recall formal systems for the logics and then establish the biequivalences between the syntax and the semantics in §3. Finally, we show that finitely presentable strategies suffices for PCF in §4.

Convention. We employ the following conventions:

- We write $\wp(X)$ for the *power set* of a set X, and $|X|$ for the *cardinality* of X;
- We use bold small letters $\boldsymbol{s}, \boldsymbol{t}, \boldsymbol{u}, \boldsymbol{v}$, etc. for sequences, in particular $\boldsymbol{\epsilon}$ for the *empty sequence*, and small letters a, b, m, n, x, y, etc. for elements of sequences;
- We write X^* for the set of all finite sequences of elements of X, and given a map $f : X \to Y$ we write $f^* : X^* \to Y^*$ for its free-monoid map $x_1 x_2 \dots x_n \mapsto f(x_1)f(x_2)\dots f(x_n)$;
- We define $\overline{n} := \{ 1, 2, \dots, n \}$ for each $n \in \mathbb{N}_+ := \mathbb{N} \setminus \{0\}$, and $\overline{0} := \emptyset$;

- We write $x_1 x_2 \dots x_{|\boldsymbol{s}|}$ for a sequence $\boldsymbol{s} = (x_1, x_2, \dots, x_{|\boldsymbol{s}|})$, with $x^n := \underbrace{xx\dots x}_{n}$, where $|\boldsymbol{s}|$ is the number of elements or *length* of $\boldsymbol{s}$, and define $\boldsymbol{s}(i) := x_i$ $(i \in \overline{|\boldsymbol{s}|})$ and $\boldsymbol{s}_{\leqslant i} := x_1 x_2 \dots x_i$;
- If L is a set of finite sequences, then $\underline{L} := \bigcup_{\boldsymbol{s} \in L} \underline{\boldsymbol{s}}$, where $\underline{\boldsymbol{s}} := \{\, \boldsymbol{s}(i) \mid i \in \overline{|\boldsymbol{s}|} \,\}$;
- A *concatenation* of finite sequences $\boldsymbol{s}$ and $\boldsymbol{t}$ is represented by their juxtaposition $\boldsymbol{st}$ (or $\boldsymbol{s}.\boldsymbol{t}$), but we often write $a\boldsymbol{s}$, $\boldsymbol{t}b$, $\boldsymbol{u}c\boldsymbol{v}$ for $(a)\boldsymbol{s}$, $\boldsymbol{t}(b)$, $\boldsymbol{u}(c)\boldsymbol{v}$, and so on;
- We write $\mathrm{Even}(\boldsymbol{s})$ (respectively, $\mathrm{Odd}(\boldsymbol{s})$) if $|\boldsymbol{s}|$ is even (respectively, odd), and define $S^{\mathrm{P}} := \{\, \boldsymbol{s} \in S \mid \mathrm{P}(\boldsymbol{s}) \,\}$ for a set S of sequences and a predicate $\mathrm{P} \in \{\mathrm{Even}, \mathrm{Odd}\}$;
- We write $\boldsymbol{s} \preceq \boldsymbol{t}$ if $\boldsymbol{s}$ is a *prefix* of a sequence $\boldsymbol{t}$, and given a set S of sequences, $\mathrm{Pref}(S)$ for the set of all prefixes of sequences in S, i.e., $\mathrm{Pref}(S) := \{\, \boldsymbol{s} \mid \exists \boldsymbol{t} \in S.\, \boldsymbol{s} \preceq \boldsymbol{t} \,\}$.

2 Combinatorial arenas and finitely presentable strategies

This section introduces two central concepts of the present work, *combinatorial arenas* (§2.1–2.3) and *finitely presentable strategies* (§2.4), and studies their basic properties. These combinatorial structures recast dynamic game semantics [YA20], yet requiring nothing infinite, nonatomic or extrinsic to them, and constitute our basis. We conclude the present section with bicategories of combinatorial arenas and finitely presentable strategies for logic and computation (§2.5).

2.1 Combinatorial arenas

We begin with a review of some basic concepts in graph theory that are necessary for defining combinatorial arenas. Some of them can be infinitary, but we later focus on finitary ones.

An ***edge*** is any two-element set $\{x, y\}$, and it is said to be ***on*** a set S if $x, y \in S$. A ***(simple) graph*** is a pair $G = (\mathscr{V}_G, \mathscr{E}_G)$ of a set $\mathscr{V}_G$, whose elements are called ***vertices***, and a set $\mathscr{E}_G$ of edges on $\mathscr{V}_G$. A graph G is said to be ***finite*** (respectively, ***empty***) if so is $\mathscr{V}_G$, ***null*** if $\mathscr{E}_G = \emptyset$, and ***complete*** if $\{x, y\} \in \mathscr{E}_G$ for all $x, y \in \mathscr{V}_G$ with $x \neq y$. A graph H is called a ***subgraph*** of G, written $H \sqsubseteq G$, if $\mathscr{V}_H \subseteq \mathscr{V}_G$ and $\mathscr{E}_H \subseteq \mathscr{E}_G$. We always assume $\mathscr{V}_G \cap \wp(\mathscr{V}_G) = \emptyset$.

A ***path*** in G is a finite sequence $v_0 v_1 \dots v_n \in \mathscr{V}_G^*$ of pairwise distinct vertices with $\{v_i, v_{i+1}\} \in \mathscr{E}_G$ for $i = 0, 1, \dots, n-1$. A nonempty graph is said to be ***connected*** if it has a path between each pair of vertices. If $\boldsymbol{s} = v_0 v_1 \dots v_{n-1}$ is a path in G with $n \geqslant 3$, then $\boldsymbol{s}v_0$ is called a ***cycle*** in G. A graph is said to be ***acyclic*** if it does not contain a cycle.

A ***tree*** is a connected, acyclic graph, or equivalently a graph with exactly one path between each pair of vertices. A ***rooted tree*** is a tree T together with a distinguished vertex rt_T, called the ***root***, and a ***rooted forest*** is a disjoint union F of rooted trees. The partial order $\leqslant_F$ on $\mathscr{V}_F$ defines $x \leqslant_F y$ if x is in the path from a root to y. If $p \leqslant_F c$ and $\{p, c\} \in \mathscr{E}_F$, then p is called a ***parent*** of c, or c a ***child*** of p, in F, where $p \rightarrow_F c$ or $p \rightarrow c$ denotes the edge $e := \{p, c\}$ with $\mathrm{src}_F(e) := p$ and $\mathrm{tgt}_F(e) := c$ called its ***source*** and ***target***, respectively. A ***leaf*** of F is a vertex of F with no children.[1] Two vertices of F are called ***siblings*** in F if they are both roots or have the same parent, and a subset $S \subseteq \mathscr{V}_F$ is said to be ***fraternal*** in F if $S \neq \emptyset$ and its elements are pairwise siblings in F. The ***depth*** $\mathrm{dep}_F(x)$ of $x \in \mathscr{V}_F$ is the length of the path from a root to x, lifted to $f \in \mathscr{E}_F$ by $\mathrm{dep}_F(f) := \mathrm{dep}_F(\mathrm{src}_T(f))$, and to F by $\mathrm{dep}(F) := \sup(\{\, \mathrm{dep}_F(x) \mid x \in \mathscr{V}_F \,\})$.

Let $\mathscr{V}_F^i \subseteq \mathscr{V}_F$ $(i \in \{0\} \cup \overline{\mathrm{dep}(F)})$ and $\mathscr{E}_F^{j-1} \subseteq \mathscr{E}_F$ $(j \in \overline{\mathrm{dep}(F)})$ consist of elements of depth i and $j-1$, respectively; let $\mathscr{V}_F^{\geqslant i} := \bigcup_{k=i}^{\mathrm{dep}(F)} \mathscr{V}_F^k$ and $\mathscr{E}_F^{\geqslant j-1} := \bigcup_{l=j-1}^{\mathrm{dep}(F)-1} \mathscr{E}_F^l$.

[1] A leaf can be a *root* with no children. We adopt this definition for convenience; e.g., see Definition 2.3.

Notation. For readability, we omit *tags* for disjoint union $\uplus$. For instance, given sets A and B, we write $x \in A \uplus B$ if $x \in A$ or $x \in B$; also, if E and F are respectively sets of edges on A and B, then we write $E \uplus F$ for the disjoint union of E and F whose elements are edges on $A \uplus B$.

We represent a rooted forest by the triple $F = (\mathscr{R}_F, \mathscr{V}_F, \mathscr{E}_F)$ of the set $\mathscr{R}_F$ of all roots, the set $\mathscr{V}_F$ of all vertices and the set $\mathscr{E}_F$ of all edges. If $\mathscr{R}_F$ is a singleton set $\mathscr{R}_F = \{\mathrm{rt}_F\}$, then we abbreviate it as rt_F. We simply write $\mathscr{R}_F$ for F if $\mathscr{R}_F = \mathscr{V}_F$ and $\mathscr{E}_F = \emptyset$. We also depict a rooted forest in the usual diagrammatic form, where edges are written $\rightarrow$ or $\Rightarrow$.

Given rooted forests F and G, an element $\star$ and sets X and Y such that the intersection $X_F := X \cap \mathscr{V}_F$ is fraternal in F, i.e., the set $\mathrm{prt}(X_F) := \bigcup_{x \in X_F} \{\, p \in \mathscr{V}_F \mid p \rightarrow_F x \,\}$ of all parents of elements of X_F is empty or singleton, we define the following rooted forests:

- $\star.F := (\star, \{\star\} \uplus \mathscr{V}_F, \{\, \star \rightarrow r \mid r \in \mathscr{R}_F \,\} \uplus \mathscr{E}_F)$, i.e., $\star.F$ is the rooted tree obtained from F by disjointly adding the element $\star$ as the new root;
- $F \cup G := (\mathscr{R}_F \cup \mathscr{R}_G, \mathscr{V}_F \cup \mathscr{V}_G, \mathscr{E}_F \cup \mathscr{E}_G)$, i.e., $F \cup G$ is the *union* of F and G;
- $F \uplus G := (\mathscr{R}_F \uplus \mathscr{R}_G, \mathscr{V}_F \uplus \mathscr{V}_G, \mathscr{E}_F \uplus \mathscr{E}_G)$, i.e., $F \uplus G$ is the *disjoint union* of F and G;
- $F_X := (X_F, \bigcup_{x_0 \in X_F} \{\, x \in \mathscr{V}_F \mid x_0 \leqslant_F x \,\}, \bigcup_{x_0 \in X_F} \{\, e \in \mathscr{E}_F \mid x_0 \leqslant_F \mathrm{src}_F(e) \,\})$, i.e., F_X is the largest rooted *subforest* (i.e., a subgraph that is a forest) of F with roots in X_F;
- $F_X^\bullet := F_X$ if $\mathrm{prt}(X_F) = \emptyset$, and $F_X^\bullet := r.F_X$ if $\mathrm{prt}(X_F) = \{r\}$;
- $F{\upharpoonright}_Y := (\{\, y \in Y_F \mid \forall y_0 \in Y_F.\, y_0 \not\rightarrow_F y \,\}, Y_F, \mathscr{E}_F \cap \wp(Y_F))$, i.e., $F{\upharpoonright}_Y$ is the rooted subforest of F induced by Y_F.

Next, recall that a ***partition*** of a finite set S is a (necessarily finite) set $\{S_i\}_{i \in \overline{n}}$ of nonempty subsets $S_i \subseteq S$ such that $\bigcup_{i=1}^n S_i = S$ and $S_i \cap S_j = \emptyset$ for all $i, j \in \overline{n}$. We generalise a partition to a ***recursive partition*** of S, which is a (necessarily finite) rooted tree P such that

1. The root is S, and other vertices are nonempty subsets of S;
2. If a vertex V has children V_i ($i \in \overline{n}$), then the set $\{V_i\}_{i \in \overline{n}}$ is a partition of V with $n > 1$.

The recursive partition P is said to be ***empty*** if $S = \emptyset$, ***trivial*** if $S \neq \emptyset$ and $\mathrm{dep}(P) = 0$, and ***exhaustive*** if its leaves are all singleton sets.

Having recalled these preliminary concepts, let us proceed to the central concept of combinatorial arenas (Definition 2.7). Roughly, a combinatorial arena is a finite rooted forest equipped with three combinatorial structures that correspond respectively to *multiplicatives*, *additives* and *exponentials* in linear logic [GL87]. We first introduce these auxiliary structures.

First, the structure for multiplicatives is to specify a class of finite sequences of vertices of the underlying finite rooted forest, which are to serve as *moves* in a game:

Definition 2.1 (multiplicative structures)**.** A ***multiplicative structure*** on a (necessarily finite) rooted forest F is a finite rooted forest μ with $\mathscr{V}_\mu = \mathscr{V}_F$ such that the set $\underline{\boldsymbol{s}}$ for each nonempty path $\boldsymbol{s}$ in μ is fraternal in F. A vertex of F is said to be ***switching*** in μ if it is not a leaf in μ.

Example 2.2. Consider the finite rooted forest

$$\begin{tikzcd}
 & x \arrow[dl] \arrow[d] & y \arrow[d] & z & s & t \\
u & v & w & & &
\end{tikzcd}$$

for which we write F. For instance, the following two finite rooted forests

$$\begin{tikzcd}
x \arrow[d] & z \arrow[d] & t & u & v & w \\
y & s & & & &
\end{tikzcd}
\qquad
\begin{tikzcd}
& y \arrow[r] \arrow[dl] \arrow[d] \arrow[dr] & s & u \arrow[d] & w \\
x & z & t & v &
\end{tikzcd}$$

are both multiplicative structures on F. For a record purpose, we write μ for the right one. For convenience, we represent the pair (F, μ) by the union of F and μ with edges of μ written $\Rightarrow$ to distinguish them from those $\rightarrow$ of F, i.e.,

$$\begin{tikzcd}
& x \arrow[dl] \arrow[d] & y \arrow[l, Rightarrow] \arrow[r, Rightarrow] \arrow[rr, Rightarrow, bend right] \arrow[rrr, Rightarrow, bend left] \arrow[d] & z & s & t \\
u \arrow[r, Rightarrow] & v & w & & &
\end{tikzcd}$$

in which y and u are switching in μ. We employ this representation throughout this article.

On the other hand, for counterexamples, neither of the two graphs

$$\begin{tikzcd}
x \arrow[d] & y \arrow[d] & u \arrow[d] \\
w & t & v
\end{tikzcd}
\qquad
\begin{tikzcd}
& y \arrow[r] \arrow[dl] \arrow[d] \arrow[dr] & s & u \arrow[d] & w \\
x & t & z \arrow[ll, bend left] & v &
\end{tikzcd}$$

is a multiplicative structure on F; the left one does not satisfy the first nor the second axiom of Definition 2.1, and the right one is not even a (simple) forest.

The intuition is that a finite rooted forest F determines components of a game similarly to an *arena* [HO00] except that nonempty maximal paths in a multiplicative structure μ on F, not vertices of F, serve as ***moves*** in the game. Arenas are a class of rooted *dags*, where implication $\Rightarrow$ between arenas entails the use of dags (going beyond simple graphs). Yet, it makes our dynamic graph method introduced later much simpler to stay in rooted (simple) forests since dynamics on them can be defined in terms of that on vertices. For this reason, we recast implication via multiplicative structures in such a way that it preserves rooted forests (Definition 2.9).

More specifically, a switching vertex functions as a *hub* between the domain and the codomain of our implication (Definition 2.9), and this structure enables combinatorial arenas to stay in rooted forests under implication. Moreover, the distinction between switching and non-switching vertices is, albeit plain, responsible for not only this simple graph approach but also many of the present results, which existing game semantics in the literature could not attain, as we shall see. For instance, the second part of Example 2.26 illustrates that the distinction is essential for our combinatorial characterisation of formal systems embracing cuts or *intensionality* (§3.2).

The first axiom $\mathscr{V}_\mu = \mathscr{V}_F$ ensures that the set $\mathscr{V}_F$ has no redundancy, i.e., each vertex of F is used for a move. The second axiom facilitates the inheritance of *pointers* from HO [HO00] to our setting (Definition 2.19). Besides, this axiom guarantees that every move consists of vertices of the same depth in F. If it consists of vertices of odd-depth in F, then it is called an ***Opponent's move*** or ***O-move***, and otherwise a ***Player's move*** or ***P-move***.

Convention. We call the pair (F, μ) of a finite rooted forest F and a multiplicative structure μ on F a ***multiplicative pair***, and write $\mathcal{M}(\mu)$ for the set of all moves, i.e., nonempty maximal paths, in μ. Because we often talk about sequences of moves, we use the vector notation $\vec{m}$ for moves (instead of the bold letter $\boldsymbol{m}$) so that those sequences are written $\boldsymbol{s} = \vec{m}_1\vec{m}_1 \dots \vec{m}_n$.

Next, the structure for additives is to determine moves to be *unavailable*:

Definition 2.3 (additive structures)**.** An ***additive structure*** on a multiplicative pair (F, μ) is the (necessarily disjoint) union $\alpha = \alpha[0] \cup \alpha[2]$ of a set $\alpha[0]$ of leaves in F that are not switching in μ and a set $\alpha[2]$ of edges between siblings in μ.

Example 2.4. The set $\alpha := \{z, w, \{x, z\}, \{x, t\}, \{z, t\}\}$ is an additive structure on the multiplicative pair (F, μ) given in Example 2.2. For a counterexample, the set $\{\{x, y\}, \{y, w\}\}$ is not an additive structure on the multiplicative pair (F, μ) since y and w are not siblings in μ.

The idea of an additive structure α on a multiplicative pair (F, μ) is: Each move $\vec{m} \in \mathcal{M}(\mu)$ such that $\underline{\vec{m}} \cap \alpha[0] \neq \emptyset$ *can never be played* in a game; when a move $\vec{n} \in \mathcal{M}(\mu)$ with $x \in \underline{\vec{n}}$ and $\{x, y\} \in \alpha[2]$ is made, each move containing a vertex of $F_{\{y\}}$ becomes *unavailable*. Such graph-theoretic *dynamics* of α is to model additives in linear logic in a finitary way, where the subset $\alpha[0] \subseteq \alpha$ corresponds to 0-ary additives, and the other $\alpha[2] \subseteq \alpha$ to binary ones (Definition 2.9). We make the dynamics of α precise in §2.2. The axioms on α are for Theorem 2.13 to hold.

Finally, the structure for exponentials is to define what to be *duplicated* during a play:

Definition 2.5 (exponential structures)**.** An ***exponential structure*** on a multiplicative pair (F, μ) is a finite rooted forest ϵ whose vertices are pairs (S, i) of a fraternal set S in μ and a natural number i unique among the second elements of vertices of ϵ, and edges $(S, i) \to (T, j)$ are the superset relation $S \supseteq T$ such that the first elements of roots of ϵ are pairwise disjoint.

Convention. We use the pair (S, i), not the set S itself, as a vertex of an exponential structure because S may occur more than once. Nevertheless, as in the case of occurrences in a sequence, we usually abbreviate (S, i) as S; this convention does not bring confusion in practice.

Notation. For a vertex S of an exponential structure ϵ on a multiplicative pair (F, μ), we define $\overline{S} := \mathscr{V}_{\mu_S} = \bigcup_{x_0 \in S} \{ x \in \mathscr{V}_F \mid x_0 \leqslant_\mu x \}$.

Example 2.6. Consider again the multiplicative pair (F, μ) in Example 2.2. The rooted forest

$$\{x, z\} \to \{x\} \qquad\qquad \{v\} \to \{v\} \qquad\qquad \{w\}$$

written ϵ, is an exponential structure on (T, μ). Strictly speaking, the two occurrences of $\{v\}$ are distinguished by two distinct natural numbers attached on them. On the other hand, the rooted tree $\{u, v\} \to \{v\}$, written ϵ', is not because the vertex $\{u, v\}$ is not fraternal in μ.

To explain the idea of an exponential structure ϵ on a multiplicative pair (F, μ), let us define the finite sequence $\epsilon(x)$ of fraternal sets in μ for each $x \in \mathscr{V}_F$ by $\epsilon(x) := \epsilon(x)_1 \epsilon(x)_2 \dots \epsilon(x)_n$ if there is the longest nonempty path $\epsilon(x)_1 \supseteq \epsilon(x)_2 \supseteq \cdots \supseteq \epsilon(x)_n$ in ϵ whose elements $\epsilon(x)_j$, $1 \leqslant j \leqslant n$, all contain x, and by $\epsilon(x) := \boldsymbol{\epsilon}$ otherwise. Then, the idea is: When a move $\vec{m} \in \mathcal{M}(\mu)$ is made during a play, the rooted subforests $F_{\overline{\epsilon(x)_j}}$ of F are *duplicated* in a suitable way for each $x \in \underline{\vec{m}}$. This graph-theoretic *dynamics* of ϵ is to model exponentials in a finitary way (§2.2).

We are now ready to introduce our combinatorial reformulation of games:

Definition 2.7 (combinatorial arenas)**.** A ***combinatorial arena*** $\mathscr{A}$ is a multiplicative pair $(|\mathscr{A}|, \mu_{\mathscr{A}})$ together with an additive structure $\alpha_{\mathscr{A}}$ and an exponential structure $\epsilon_{\mathscr{A}}$ on $(|\mathscr{A}|, \mu_{\mathscr{A}})$ that admits an exhaustive recursive partition P_S of each maximal fraternal set S in $\mu_{\mathscr{A}}$ such that, for each partition $S_I = \{S_i\}_{i \in I}$ occurring in P_S, the following holds:

1. The relation $[\alpha_{\mathscr{A}}[2]]_I$ between elements $S_i, S_j \in S_I$ $(i, j \in I)$ given in terms of arbitrarily chosen representatives $x \in S_i$ and $y \in S_j$ by

   $$[\alpha_{\mathscr{A}}[2]]_I(S_i, S_j) :\Leftrightarrow \{x, y\} \in \alpha_{\mathscr{A}}[2]$$

   is well-defined, i.e., independent of the choice of the representatives x and y;

2. The graph $(S_I, [\alpha_{\mathscr{A}}[2]]_I)$ is null or complete;
3. $V \cap S_i \neq \emptyset$ implies $V \subseteq S_i$ or $S_i \subseteq V$ for all $V \in \mathscr{V}_{\epsilon_{\mathscr{A}}}$ and $i \in I$.

Vertices and ***edges*** of $\mathscr{A}$ are those of $|\mathscr{A}|$, and ***moves*** in $\mathscr{A}$ are elements of $\mathcal{M}_{\mathscr{A}} := \mathcal{M}(\mu_{\mathscr{A}})$. A vertex of $\mathscr{A}$ is said to be ***switching*** if so is it in $\mu_{\mathscr{A}}$. A combinatorial arena $\mathscr{S}$ is called a ***combinatorial subarena*** of $\mathscr{A}$ if it satisfies $|\mathscr{S}| \sqsubseteq |\mathscr{A}|$, $\mu_{\mathscr{S}} \sqsubseteq \mu_{\mathscr{A}}$, $\alpha_{\mathscr{S}} \subseteq \alpha_{\mathscr{A}}$ and $\epsilon_{\mathscr{S}} \sqsubseteq \epsilon_{\mathscr{A}}$.

The three axioms of Definition 2.7 are again for Theorem 2.13 to hold. See Example 2.8, Definition 2.9 and Theorem 2.13 on how multiplicative (respectively, additive, exponential) structures correspond to multiplicatives (respectively, additives, exponentials) in linear logic.

Example 2.8. Consider again the multiplicative pair (F, μ) given in Example 2.2 together with the additive structure α in Example 2.4 and the exponential structure ϵ in Example 2.6. They constitute a combinatorial arena, which is written $\mathscr{A}$ for a record purpose.

For counterexamples, the additive structure $\{\{x, z\}, \{z, t\}, \{s, t\}\}$ on (F, μ) does not satisfy the first two axioms of Definition 2.7, and the exponential structure on (F, μ) that consists of the unique root $\{s, t\}$ does not satisfy the third axiom with respect to the additive structure α.

Let us next introduce constructions on combinatorial arenas, which correspond to those on formulae in intuitionistic linear logic [GL87] (recalled in Definition 3.2):

Definition 2.9 (constructions on combinatorial arenas)**.** Let $\mathscr{A}$ and $\mathscr{B}$ be combinatorial arenas.

- ***Top*** is the combinatorial arena $\top := (\emptyset, \emptyset, \emptyset, \emptyset)$;
- ***One*** is the combinatorial arena $1 := (1, 1, \{1\}, \emptyset)$, where 1 is an arbitrarily fixed element;
- The ***tensorial negation***[2] of $\mathscr{A}$ is the combinatorial arena $\neg\mathscr{A}$ defined by
$$|\neg\mathscr{A}| := \neg.|\mathscr{A}| \qquad \mu_{\neg\mathscr{A}} := \neg \uplus \mu_{\mathscr{A}} \qquad \alpha_{\neg\mathscr{A}} := \alpha_{\mathscr{A}} \qquad \epsilon_{\neg\mathscr{A}} := \epsilon_{\mathscr{A}},$$
where $\neg$ is an arbitrarily fixed element;
- The ***tensor*** of $\mathscr{A}$ and $\mathscr{B}$ is the combinatorial arena $\mathscr{A} \otimes \mathscr{B}$ defined by
$$|\mathscr{A} \otimes \mathscr{B}| := |\mathscr{A}| \uplus |\mathscr{B}| \qquad \mu_{\mathscr{A} \otimes \mathscr{B}} := \mu_{\mathscr{A}} \uplus \mu_{\mathscr{B}} \qquad \alpha_{\mathscr{A} \otimes \mathscr{B}} := \alpha_{\mathscr{A}} \uplus \alpha_{\mathscr{B}} \qquad \epsilon_{\mathscr{A} \otimes \mathscr{B}} := \epsilon_{\mathscr{A}} \uplus \epsilon_{\mathscr{B}};$$
- The ***product*** or ***with*** of $\mathscr{A}$ and $\mathscr{B}$ is the combinatorial arena $\mathscr{A} \mathbin{\&} \mathscr{B}$ defined by
$$|\mathscr{A} \mathbin{\&} \mathscr{B}| := |\mathscr{A} \otimes \mathscr{B}| \qquad \mu_{\mathscr{A} \& \mathscr{B}} := \mu_{\mathscr{A} \otimes \mathscr{B}}$$
$$\alpha_{\mathscr{A} \& \mathscr{B}} := \alpha_{\mathscr{A} \otimes \mathscr{B}} \uplus \{\, \{a, b\} \mid a \in \mathscr{R}_{|\mathscr{A}|} \cap \mathscr{R}_{\mu_{\mathscr{A}}}, b \in \mathscr{R}_{|\mathscr{B}|} \cap \mathscr{R}_{\mu_{\mathscr{B}}} \,\} \qquad \epsilon_{\mathscr{A} \& \mathscr{B}} := \epsilon_{\mathscr{A} \otimes \mathscr{B}};$$
- The ***of-course*** of $\mathscr{A}$ is the combinatorial arena $!\mathscr{A}$ defined by
$$|!\mathscr{A}| := |\mathscr{A}| \qquad \mu_{!\mathscr{A}} := \mu_{\mathscr{A}} \qquad \alpha_{!\mathscr{A}} := \alpha_{\mathscr{A}}$$
$$\epsilon_{!\mathscr{A}} := \mathscr{R}_{\mathscr{A}} * \epsilon_{\mathscr{A}} := \big((\mathscr{R}_{|\mathscr{A}|} \cap \mathscr{R}_{\mu_{\mathscr{A}}}).(\epsilon_{\mathscr{A}})_{\wp(\mathscr{R}_{|\mathscr{A}|} \cap \mathscr{R}_{\mu_{\mathscr{A}}})}\big) \uplus \big(\epsilon_{\mathscr{A}} \upharpoonright_{\wp(\mathscr{V}_{|\mathscr{A}|} \setminus (\mathscr{R}_{|\mathscr{A}|} \cap \mathscr{R}_{\mu_{\mathscr{A}}}))}\big);$$

[2]This terminology comes from the naming of a similar construction on games in [MT10].

- The ***linear implication*** from $\mathscr{A}$ to $\mathscr{B}$ is the combinatorial arena $\mathscr{A} \multimap \mathscr{B}$ defined by

$$|\mathscr{A} \multimap \mathscr{B}| := \multimap.|\mathscr{A}| \uplus |\mathscr{B}| \qquad \mu_{\mathscr{A} \multimap \mathscr{B}} := \mu_{\mathscr{A}} \uplus \mu_{\mathscr{B}} \upharpoonright_{\mathscr{V}_{|\mathscr{B}|}^{\geqslant 1}} \uplus \multimap.(\mu_{\mathscr{B}} \upharpoonright_{\mathscr{R}_{|\mathscr{B}|}})$$

$$\alpha_{\mathscr{A} \multimap \mathscr{B}} := \alpha_{\mathscr{A}} \uplus \alpha_{\mathscr{B}} \qquad \epsilon_{\mathscr{A} \multimap \mathscr{B}} := \epsilon_{\mathscr{A}} \uplus \epsilon_{\mathscr{B}},$$

  where $\multimap$ is an arbitrarily fixed element, and the ***implication*** from $\mathscr{A}$ to $\mathscr{B}$ is

$$\mathscr{A} \Rightarrow \mathscr{B} := !\mathscr{A} \multimap \mathscr{B}.$$

Convention. Linear implication is right associative, while other binary operations are left associative. Each unary operation precedes all binary ones, and tensorial negation all other operations; linear implication is preceded by all other operations.

Example 2.10. The linear implication $\bot \multimap \bot \otimes \bot$ comprises of simple graphs only, while the corresponding arena [HO00] does not. Note that the switching vertex $\multimap$ makes this difference.

It is instructive to observe that switching vertices also contribute to the inequalities

$$!\top = \top = \top \otimes \top = \top \,\&\, \top \neq \top \multimap \top \neq (\top \multimap \top) \otimes (\top \multimap \top) \qquad !(\top \multimap \bot) \neq \top \multimap !\bot$$

$$\top \neq \mathscr{A} \multimap \top \neq \mathscr{B} \multimap \top \qquad \neg\mathscr{A} \neq \mathscr{A} \multimap \bot \qquad \top \multimap \mathscr{A} \neq \mathscr{A}$$

$$\top \multimap \bot \,\&\, \bot \neq \bot \,\&\, (\top \multimap \bot) \neq (\top \multimap \bot) \,\&\, (\top \multimap \bot),$$

where $\mathscr{A}$ and $\mathscr{B}$ are arbitrary combinatorial arenas such that $\mathscr{A} \neq \mathscr{B}$. These inequalities are quite remarkable because existing game semantics [AJM00, HO00, McC98] does not attain them. We shall see later that some of the inequalities are crucial for our fully complete interpretation of intuitionistic linear logic (Theorem 3.26).

Proposition 2.11 (well-defined constructions on combinatorial arenas)**.** *Top and one form combinatorial arenas, and combinatorial arenas are closed under tensorial negation, tensor, with, of-course and linear implication.*

Proof. Straightforward and left to the reader. $\square$

Example 2.12. The combinatorial arena $(F, \mu, \alpha, \epsilon)$ given in Example 2.8 can be constructed as $!1_{[w]} \multimap_{[y]} (!(1_{[z]} \,\&\, !\neg_{[x]}(\top \multimap_{[u]} !!\bot_{[v]})) \,\&\, \bot_{[t]}) \otimes \bot_{[s]}$, where the subscripts are tags to indicate the vertices corresponding to the constructions. We use this notation throughout this article.

The proof of the following proposition describes how to inductively construct a given combinatorial arena. In particular, the proof explains how the tags in Example 2.12 are chosen.

Theorem 2.13 (a free characterisation of combinatorial arenas)**.** *Every combinatorial arena $\mathscr{A}$ can be obtained, up to graph isomorphisms $\cong$ on the underlying finite rooted forest $|\mathscr{A}|$, from top and/or one by tensorial negation, linear implication, tensor, with and/or of-course.*

Proof. Let $\mathscr{A}$ be a combinatorial arena. We proceed by induction on $\mathrm{Size}(\mathscr{A}) := |\mathscr{V}_{|\mathscr{A}|}| \uplus |\mathscr{V}_{\epsilon_{\mathscr{A}}}|$.

The base case corresponds to $\mathrm{Size}(\mathscr{A}) = 0$, in which $\mathscr{A} = \top$. Hence, we are done. For the inductive steps, we henceforth assume $\mathrm{Size}(\mathscr{A}) > 0$, which implies $\mathscr{R}_{|\mathscr{A}|} \neq \emptyset$.

First, assume $\epsilon_{\mathscr{A}} = \mathscr{R}_{|\mathscr{A}|} * \epsilon'$ for some exponential structure ϵ'. Then, $\mathscr{A}' := (|\mathscr{A}|, \mu_{\mathscr{A}}, \alpha_{\mathscr{A}}, \epsilon')$ is a combinatorial arena with $\mathrm{Size}(\mathscr{A}') < \mathrm{Size}(\mathscr{A})$ and $!\mathscr{A}' = \mathscr{A}$. Thus, the induction hypothesis verifies the claim for $\mathscr{A}$. Now, we can assume $\epsilon_{\mathscr{A}} \neq \mathscr{R}_{|\mathscr{A}|} * \epsilon''$ for all exponential structures ϵ''.

Next, suppose that $\mathscr{R}_{|\mathscr{A}|}$ is singleton. If $\alpha_{\mathscr{A}} \cap \mathscr{R}_{|\mathscr{A}|} \neq \emptyset$, then $\mathscr{A} \cong 1$; thus, assume otherwise. Then, $\alpha_{\mathscr{A}} \cap \mathscr{R}_{|\mathscr{A}|} = \emptyset$ and $\epsilon_{\mathscr{A}} \neq \mathscr{R}_{|\mathscr{A}|}.\epsilon''$ for all exponential structures ϵ'', so $\mathscr{A} \cong \neg\mathscr{A}''$ for some combinatorial arena $\mathscr{A}''$. Because $\mathrm{Size}(\mathscr{A}') < \mathrm{Size}(\mathscr{A})$, the induction hypothesis on $\mathscr{A}''$ verifies the claim for $\mathscr{A}$. Hence, in the following, it suffices to suppose that $\mathscr{R}_{|\mathscr{A}|}$ is not singleton.

Lastly, consider the case where $\mathscr{R}_{|\mathscr{A}|}$ is not singleton. The rooted subforest $\mu' := \mu_{\mathscr{A}} \upharpoonright_{\mathscr{R}_{|\mathscr{A}|}}$ of $\mu_{\mathscr{A}}$ then has more than one vertex. We then proceed by case analysis on μ'. If μ' is a rooted tree, where we write x_0 for its unique root, then we define

$$V_1 := \{\, x \in \mathscr{V}_{|\mathscr{A}|} \mid x_0 \rightarrow_{|\mathscr{A}|} x \,\} \qquad\qquad V_2 := \mathscr{R}_{|\mathscr{A}|} \setminus \{x_0\}.$$

Let $\mathscr{A}_1$ be the combinatorial subarena of $\mathscr{A}$ induced by $|\mathscr{A}_1| := |\mathscr{A}|_{V_1}$, and $\mathscr{A}_2$ by $|\mathscr{A}_2| := |\mathscr{A}|_{V_2}$. Clearly, $\mathscr{A} \cong \mathscr{A}_1 \multimap \mathscr{A}_2$ and $\mathrm{Size}(\mathscr{A}_i) < \mathrm{Size}(\mathscr{A})$ $(i = 1, 2)$. Hence, the induction hypotheses on $\mathscr{A}_i$ prove the claim for $\mathscr{A}$. It remains to assume that μ' is not a rooted tree. In this case, there is a nonempty, nontrivial, exhaustive recursive partition P of $\mathscr{R}_{\mu'}$ that satisfies the three axioms of Definition 2.7. Let $S_I = \{S_i\}_{i \in I}$ be $\mathscr{V}_P^1$, where $n := |I| > 1$. The graph $(S_I, [\alpha_{\mathscr{A}}[2]]_I)$ is well-defined and either null or complete by the first two axioms. If it is complete, then $\mathscr{A} \cong \&_{j=1}^n \mathscr{A}'_j$ for some combinatorial arenas $\mathscr{A}'_j$ with $\mathrm{Size}(\mathscr{A}'_j) < \mathrm{Size}(\mathscr{A})$. If it is null, then $\mathscr{A} \cong \otimes_{j=1}^n \mathscr{A}'_j$. In either case, the induction hypotheses on $\mathscr{A}'_j$ deduce the claim for $\mathscr{A}$ by the third axiom. □

This result is the first step towards our biequivalences between logic and combinatorics (§3.2). In addition, this free characterisation provides us with a concise representation for an arbitrary combinatorial arena in terms of the inductive constructions. This representation, combined with the tags $(\_)_{[\_]}$ introduced in Example 2.12, is useful because it can be quite intricate and tedious to directly describe a combinatorial arena as seen in previous examples.

At the end of this section, we show that combinatorial arenas form *Hopf algebras* [Hop64]:

Proposition 2.14 (combinatorial Hopf algebras)**.** *Combinatorial arenas yield Hopf algebras.*

Proof (sketch). Schmitt [Sch93, §3] proved that the vector space over any commutative ring with identity, whose basis consists of all finite rooted forests, gives rise to a Hopf algebra through what is called the *restriction* operation. The restriction operation calculates the rooted subforest F_X of a given finite rooted forest F induced by a given subset X of the vertex set. The claim of the theorem follows because the restriction operation easily extends to combinatorial arenas.

Schmitt presented another way of producing a Hopf algebra of finite rooted forests through the *quotient* operation [Sch93, §4], and this method is applicable to combinatorial arenas as well. Thus, we have obtained two different Hopf algebras induced by combinatorial arenas. □

To the best of our knowledge, this proposition is the first bridge between game semantics and Hopf algebras. The literature of combinatorial Hopf algebras has shown that the results and the methods of Hopf algebras are quite useful for the study of combinatorial structures. We leave it as future work to apply Hopf algebras to logic and computation via combinatorial arenas.

Finally, let us demonstrate that Proposition 2.14 does not hold for arenas [HO00]. Recall that an axiom on an arena requires that all paths from a root to a vertex in the arena have the same length. However, the restriction operation does not respect this axiom, where the problem is that arenas are not simple (directed) graphs. For instance, consider the arena $x \rightarrow y \rightarrow z \leftarrow y' \leftarrow x'$. Its subgraph induced by the set $\{x, y, y', z\}$ is the one $x \rightarrow y \rightarrow z \leftarrow y'$, which is no longer an arena because the path from x to z and the one from y' to z have different lengths.

2.2 Positions

Our next step is to adapt possible developments or *positions* in a game to combinatorial arenas. HO [HO00] defines positions to be a class of finite walks on an arena in a *finitary* fashion. These positions are, however, too coarse to interpret linear logic. In contrast, other variants of games

[AJ94, AJM00] possess finer controls on positions at the cost of carrying an (often) *infinite* set of positions as part of a game. The idea of a combinatorial arena $\mathscr{A}$ is to overcome this dilemma by allowing $\mathscr{A}$ to *vary along the growth of a position* in such a way that the positions compatible with (or *filtered* by) such graph-theoretic *dynamics* interpret linear logic in a finitary way.

The aim of this section is to make this idea precise. Let us first define the dynamics of the *binary part* $\alpha_{\mathscr{A}}[2]$ of the additive structure $\alpha_{\mathscr{A}}$. This dynamics provides $\mathscr{A}$ with a finer control on positions to model binary additives in linear logic without referring to infinite sets of positions.

Definition 2.15 (additive dynamics)**.** The ***(binary) additive dynamics*** of a combinatorial arena $\mathscr{A}$ on a vertex $x \in \mathscr{V}_{|\mathscr{A}|}$ is the subset $\mathscr{A}_\alpha(x) \subseteq \mathscr{V}_{|\mathscr{A}|}$ defined by

$$\mathscr{A}_\alpha(x) := \begin{cases} \{\, x' \in \mathscr{V}_{|\mathscr{A}|} \mid \{x, x'\} \in \alpha_{\mathscr{A}}[2] \,\} & \text{if } x \text{ is switching in } \mu_{\mathscr{A}}; \\ \{x\} \cup \{\, x' \in \mathscr{V}_{|\mathscr{A}|} \mid \{x, x'\} \in \alpha_{\mathscr{A}}[2] \,\} & \text{otherwise,} \end{cases}$$

and it is lifted to a move $\vec{m} \in \mathcal{M}_{\mathscr{A}}$ by $\mathscr{A}_\alpha(\vec{m}) := \bigcup_{x \in \underline{\vec{m}}} \mathscr{A}_\alpha(x)$.

This structure given by $\alpha_{\mathscr{A}}[2]$ yields the following graph-theoretic *dynamics*: When a move $\vec{m}$ is made during a play in $\mathscr{A}$, moves $\vec{n}$ with $\underline{\vec{n}} \cap \mathscr{A}_\alpha(\vec{m}) \neq \emptyset$ become *unavailable* (Definition 2.19), where a switching vertex never makes itself unavailable because it must be always there, unless made unavailable by another vertex, to bridge the domain and the codomain of linear implication. This dynamics matches the intuition on with: selecting one of two sides. The *0-ary part* $\alpha_{\mathscr{A}}[0]$ has nothing to do with any dynamics of $\mathscr{A}$, and its role is defined later (Definition 2.22).

Let us next formulate the graph-theoretic dynamics of the exponential structure $\epsilon_{\mathscr{A}}$, which is to model exponentials in a finitary fashion. To this end, we need some notations:

Convention. Let $\mathscr{A}$ be a combinatorial arena, and F a finite rooted forest.

- If $\epsilon_{\mathscr{A}}$ has a nonempty path whose elements all contain $x \in \mathscr{V}_{|\mathscr{A}|}$, then $\epsilon_{\mathscr{A}}(x) := (\epsilon_{\mathscr{A}}(x)_i)_{i \in \overline{n}}$, where $\epsilon_{\mathscr{A}}(x)_1 \supseteq \epsilon_{\mathscr{A}}(x)_2 \supseteq \cdots \supseteq \epsilon_{\mathscr{A}}(x)_n$ is the longest one, and otherwise $\epsilon_{\mathscr{A}}(x) := \boldsymbol{\epsilon}$;
- Given an element x and sequences $\boldsymbol{y} = y_1 y_2 \dots y_n$ and $\boldsymbol{j} = j_1 j_2 \dots j_n$, let $x\{\boldsymbol{y}; \boldsymbol{j}\}$ denote the nested pair $(\dots((x, y_1^{j_1}), y_2^{j_2}), \dots, y_n^{j_n})$, where $y_i^{j_i} := (y_i, j_i)$; it is lifted to a set X by $X\{\boldsymbol{y}; \boldsymbol{j}\} := \{\, x\{\boldsymbol{y}; \boldsymbol{j}\} \mid x \in X \,\}$, and to a finite sequence $\boldsymbol{s}$ by $\boldsymbol{s}\{\boldsymbol{y}; \boldsymbol{j}\} := (\boldsymbol{s}(i)\{\boldsymbol{y}; \boldsymbol{j}\})_{i \in \overline{|\boldsymbol{s}|}}$;
- Because $x\{\boldsymbol{\epsilon}; \boldsymbol{\epsilon}\} = x$, we suppose that each vertex of $\mathscr{A}$ is of the form $x\{\boldsymbol{y}; \boldsymbol{j}\}$; without loss of generality, we also assume $x\{\boldsymbol{y}; \boldsymbol{j}\} \neq x'$ if $x, x' \in \mathscr{V}_{|\mathscr{A}|}$, $\boldsymbol{y} \in \mathscr{V}_{|\mathscr{A}|}^*$, $\boldsymbol{j} \in \mathbb{N}_+^*$ and $|\boldsymbol{y}| > 0$;
- Given a subset $S \subseteq \mathscr{V}_F$, let $F[S\{\boldsymbol{y}; \boldsymbol{j}\}]$ denote the finite rooted forest obtained from F by replacing each vertex x of F that is in S with the element $x\{\boldsymbol{y}; \boldsymbol{j}\}$.

Then, the dynamics of $\epsilon_{\mathscr{A}}$ is that, when a move $\vec{m}$ is made during a play, the combinatorial subarenas of $\mathscr{A}$ specified by the elements of the sequence $\epsilon_{\mathscr{A}}(x)$ for all $x \in \underline{\vec{m}}$ are *duplicated* and *disjointly adjoined* to $\mathscr{A}$. This dynamics matches the intuition on of-course: producing countably many copies of formulae. Note that $\epsilon_{\mathscr{A}}(x)$ corresponds to a finite sequence of nested occurrences of of-course in a formula, e.g., $!(A \otimes (!B \,\&\, C))$. For computation on those copies of combinatorial subarenas, we make tags on the disjoint unions precise; we use $\_\{\boldsymbol{y}; \boldsymbol{j}\}$ for this aim.

Definition 2.16 (exponential dynamics)**.** Given a combinatorial arena $\mathscr{A}$, assume $y' \in \mathscr{V}_{|\mathscr{A}|}$, $\tilde{y} = y'\{\boldsymbol{y}; \boldsymbol{j}\} \in \mathscr{V}_{|\mathscr{A}|}$ and $j' \in \overline{|\epsilon_{\mathscr{A}}(\tilde{y})|}$. Let $\mathscr{A}_\epsilon(\tilde{y}, j')$ be the combinatorial arena obtained from $\mathscr{A}$ by disjointly adding its combinatorial subarena induced by the set $\overline{\epsilon_{\mathscr{A}}(\tilde{y})(j')}$, or

- $|\mathscr{A}_\epsilon(\tilde{y}, j')| := |\mathscr{A}| \cup (|\mathscr{A}|^\bullet[\mathscr{V}_{|\mathscr{A}|^\flat}\{y'; j'\}])$ $(|\mathscr{A}|^\bullet := |\mathscr{A}|^\bullet_{\overline{\epsilon_{\mathscr{A}}(\tilde{y})(j')}}$ and $|\mathscr{A}|^\flat := |\mathscr{A}|_{\overline{\epsilon_{\mathscr{A}}(\tilde{y})(j')}})$;

- $\mu_{\mathscr{A}_\epsilon(\tilde{y},j')} := \mu_{\mathscr{A}} \cup (\mu_{\mathscr{A}} \restriction_{\mathscr{V}_{|\mathscr{A}|^\flat}})[\mathscr{V}_{|\mathscr{A}|^\flat}\{y';j'\}]$;
- $\alpha_{\mathscr{A}_\epsilon(\tilde{y},j')} := \alpha_{\mathscr{A}} \cup (\alpha_{\mathscr{A}}[0] \cap \mathscr{V}_{|\mathscr{A}|^\flat})\{y';j'\} \cup \{\, e\{y';j'\} \mid e \in \alpha_{\mathscr{A}}[2] \cap \wp(\mathscr{V}_{|\mathscr{A}|^\flat})\,\}$;
- $\epsilon_{\mathscr{A}_\epsilon(\tilde{y},j')} := \epsilon_{\mathscr{A}} \cup (\epsilon_{\mathscr{A}})_{\epsilon_{\mathscr{A}}(\tilde{y})(j')}[(\mathscr{V}_{\epsilon_{\mathscr{A}}} \cap \wp(\mathscr{V}_{|\mathscr{A}|^\flat}))\{y';j'\}]$, where unlike $|\mathscr{A}|^\bullet$ or $|\mathscr{A}|^\flat$ defined above $\epsilon_{\mathscr{A}}(\tilde{y})(j')$ in $\epsilon_{\mathscr{A}_\epsilon(\tilde{y},j')}$ serves not as a set but as a vertex of $\epsilon_{\mathscr{A}}$.

Further, let $\mathscr{A}_\epsilon(\tilde{y}) := \mathscr{A}_\epsilon(\tilde{y},1)_\epsilon(\tilde{y},2)_\epsilon \dots (\tilde{y},|\epsilon_{\mathscr{A}}(\tilde{y})|)$ if $|\epsilon_{\mathscr{A}}(\tilde{y})| > 0$, and $\mathscr{A}_\epsilon(\tilde{y}) := \mathscr{A}$ otherwise.

The ***exponential dynamics*** of $\mathscr{A}$ on a move $\vec{m} \in \mathscr{M}_{\mathscr{A}}$ is the combinatorial arena

$$\mathscr{A}_\epsilon(\vec{m}) := \begin{cases} \mathscr{A}_\epsilon(\vec{m}(1))_\epsilon(\vec{m}(2))_\epsilon \dots (\vec{m}(|\vec{m}|)) & \text{if } |\vec{m}| > 0; \\ \mathscr{A} & \text{otherwise.} \end{cases}$$

Example 2.17. Recall the combinatorial arena $\mathscr{A}$ defined in Example 2.8. For understanding the dynamics of $\mathscr{A}$ described below in terms of its syntactic counterpart, we strongly recommend the reader to refer to the inductive construction of $\mathscr{A}$ given in 2.12. First, observe the dynamics of $\mathscr{A}$ on the move $yx \in \mu_{\mathscr{A}}$: The additive dynamics $\mathscr{A}_\alpha(yx)$ is the set $\{x,z,t\}$, and the exponential dynamics $\mathscr{A}_\epsilon(yx)$ is the combinatorial arena that consists of the multiplicative pair

$$\begin{tikzcd}
(u,x^1) \arrow[d, Rightarrow] & (x,x^1) \arrow[l] \arrow[dl] & (z,x^1) & (x,x^2) \arrow[r] \arrow[dr] & (u,x^2) \arrow[d, Rightarrow] \\
(v,x^1) & x \arrow[dl] \arrow[d] & y \arrow[l, Rightarrow] \arrow[ul, Rightarrow] \arrow[u, Rightarrow] \arrow[ur, Rightarrow] \arrow[r, Rightarrow] \arrow[d] \arrow[dr, Rightarrow] \arrow[drr, Rightarrow] & z & (v,x^2) \\
u \arrow[r, Rightarrow] & v & w & s & t
\end{tikzcd}$$

together with the additive structure

$$\alpha_{\mathscr{A}} \cup \{\{(x,x^1),(z,x^1)\}\}$$

and the exponential structure

$$\begin{tikzcd}
\{x,z\} \arrow[dd] & \{(x,x^1),(z,x^1)\} \arrow[dd] & & \{u\} \arrow[dd] & \{(u,x^1)\} \arrow[dd] & \{(u,x^2)\} \arrow[dd] & \\
& & \{(x,x^2)\} & & & & \{w\} \\
\{x\} & \{(x,x^1)\} & & \{v\} & \{(v,x^1)\} & \{(v,x^2)\} &
\end{tikzcd}$$

Next, the additive dynamics $\mathscr{A}_\epsilon(yx)_\alpha(w)$ of $\mathscr{A}_\epsilon(yx)$ on the move w is the singleton set $\{w\}$, and the exponential dynamics $\mathscr{A}_\epsilon(yx)_\epsilon(w)$ does nothing, i.e., $\mathscr{A}_\epsilon(yx)_\epsilon(w) = \mathscr{A}_\epsilon(yx)$.

Finally, consider the dynamicss of $\mathscr{A}_\epsilon(yx)_\epsilon(w)$ on the move $y(z,x^1)$: The additive dynamics $\mathscr{A}_\epsilon(yx)_\epsilon(w)_\alpha(y(z,x^1))$ is the set $\{(x,x^1),(z,x^1)\}$, and the exponential dynamics $\mathscr{A}_\epsilon(yx)_\epsilon(w)_\epsilon(y(z,x^1))$ is the combinatorial arena that consists of the multiplicative pair

$$\begin{tikzcd}
(u,x^1) \arrow[d, Rightarrow] & (x,x^1) \arrow[l] \arrow[dl] & (z,x^1) & (x,x^2) \arrow[r] \arrow[dr] & (u,x^2) \arrow[d, Rightarrow] \\
(v,x^1) & x \arrow[dl] \arrow[d] & y \arrow[l, Rightarrow] \arrow[ul, Rightarrow] \arrow[u, Rightarrow] \arrow[ur, Rightarrow] \arrow[r, Rightarrow] \arrow[d] \arrow[dr, Rightarrow] \arrow[drr, Rightarrow] \arrow[ddl, Rightarrow] \arrow[ddr, Rightarrow] & z & (v,x^2) \\
u \arrow[r, Rightarrow] & v & w & s & t \\
((u,x^1),z^1) \arrow[rr, Rightarrow, bend right] & ((x,x^1),z^1) \arrow[l] \arrow[r] & ((v,x^1),z^1) & ((z,x^1),z^1) &
\end{tikzcd}$$

together with the additive structure

$$\alpha_{\mathscr{A}_\epsilon(yx)_\epsilon(w)} \cup \{\{((x, x^1), z^1), ((z, x^1), z^1)\}\}$$

and the exponential structure

$$\begin{tikzcd}
\{x,z\} \arrow[d] & \{(x,x^1),(z,x^1)\} \arrow[d] & \{((x,x^1),z^1),((z,x^1),z^1)\} \arrow[d] & \\
\{x\} & \{(x,x^1)\} & \{((x,x^1),z^1)\} & \{(x,x^2)\}
\end{tikzcd}$$

$$\begin{tikzcd}
\{u\} \arrow[d] & \{(u,x^1)\} \arrow[d] & \{(u,x^2)\} \arrow[d] & \{((u,x^1),z^1)\} \arrow[d] & \\
\{v\} & \{(v,x^1)\} & \{(v,x^2)\} & \{((v,x^1),z^1)\} & \{w\}
\end{tikzcd}$$

Because these structures easily become larger and larger during a play, one may wonder if the present framework is hard to manage. Fortunately, it is not the case: One can always represent the structures concisely by suitable *syntactic sugars*; e.g., see Example 2.24.

This example also illustrates the following:

Definition 2.18 (extended vertices and moves)**.** An ***extended vertex*** of a combinatorial arena $\mathscr{A}$ is a nested pair $x'\{\boldsymbol{x}; \boldsymbol{i}\}$ with $x' \in \mathscr{V}_{|\mathscr{A}|}$, $\boldsymbol{x} \in \mathscr{V}_{|\mathscr{A}|}^*$ and $\boldsymbol{i} \in \mathbb{N}_+^*$, where x' is called its ***core***, and an ***extended O-*** (respectively, ***P-***) ***move*** in $\mathscr{A}$ is a sequence $x'_1\{\boldsymbol{x}_1; \boldsymbol{i}_1\}x'_2\{\boldsymbol{x}_2; \boldsymbol{i}_2\} \dots x'_n\{\boldsymbol{x}_n; \boldsymbol{i}_n\}$ of extended vertices of $\mathscr{A}$ such that $x'_1 x'_2 \dots x'_n \in \mathcal{M}_{\mathscr{A}}$ is an O- (respectively, P-) move in $\mathscr{A}$.

Convention. An ***extended move*** in a combinatorial arena $\mathscr{A}$ refers to an extended O- or P-move in $\mathscr{A}$. We write $\mathscr{V}_{|\mathscr{A}|}^+$ for the set of all extended vertices of $\mathscr{A}$, and $\mathcal{M}_{\mathscr{A}}^+$ for the set of all extended moves in $\mathscr{A}$. A tag of the form $\_\{\boldsymbol{y}; \boldsymbol{j}\}$ is called an ***exponential tag***. If a vertex $x' \in \mathscr{V}_{|\mathscr{A}|}$ is switching, then any extended one of the form $x'\{\boldsymbol{x}; \boldsymbol{i}\}$ is said to be ***switching*** too.

Next, recall that a position in an arena is a finite sequence of vertices equipped with a *pointer* [HO00]. Such a sequence together with a pointer is called a *justified sequence.* The idea is that each non-initial element of a justified sequence is made for a specific previous one, and a pointer represents this relation. This idea traces back to Coquand [Coq95]. An advantage of pointers is that they enable an arena to define its positions in a *finitary* way unlike other variants of games.

Let us then adapt justified sequences to combinatorial arenas:

Notation. Given an extended vertex $x = x_0\{\boldsymbol{x}; \boldsymbol{i}\}$, we write $x^\sharp$ for its core, i.e., $x^\sharp := x_0$, and this operation $(\_)^\sharp$ extends to extended moves in the evident way.

Definition 2.19 (justified sequences)**.** A ***justified sequence*** in a combinatorial arena $\mathscr{A}$ is a pair $\boldsymbol{s} = (\boldsymbol{s}, J_{\boldsymbol{s}})$ of a finite sequence $\underline{\boldsymbol{s}}$ of extended moves in $\mathscr{A}$ and a map $J_{\boldsymbol{s}} : \overline{|\boldsymbol{s}|} \to \{0\} \cup \overline{|\boldsymbol{s}| - 1}$ such that $0 \leqslant J_{\boldsymbol{s}}(i) < i$ for all $i \in \overline{|\boldsymbol{s}|}$, called the ***pointer***, that satisfies

1. If $\boldsymbol{t}\vec{m} \preceq \boldsymbol{s}$, then $\vec{m} \in \mathcal{M}_{\mathscr{A}_\epsilon(\boldsymbol{t})}$, where $\mathscr{A}_\epsilon(\boldsymbol{t}) := \mathscr{A}_\epsilon(\boldsymbol{t}(1))_\epsilon(\boldsymbol{t}(2))_\epsilon \dots (\boldsymbol{t}(|\boldsymbol{t}|))$ if $|\boldsymbol{t}| > 0$, and $\mathscr{A}_\epsilon(\boldsymbol{t}) := \mathscr{A}$ otherwise, and each $x \in \underline{\vec{m}}$ has the shortest exponential tag among all $y \in \mathscr{V}_{|\mathscr{A}_\epsilon(\boldsymbol{t})|}$ such that $y^\sharp = x^\sharp$;

2. If $i \in |\boldsymbol{s}|$, then $J_{\boldsymbol{s}}(i) = 0$ implies that $\boldsymbol{s}(i)$ consists of roots of $|\mathscr{A}_\epsilon(\boldsymbol{s}_{\leqslant i})|$, and $J_{\boldsymbol{s}}(i) > 0$ that there is $x \in \underline{\boldsymbol{s}(J_{\boldsymbol{s}}(i))}$ such that $x \to_{|\mathscr{A}_\epsilon(\boldsymbol{s}_{\leqslant i})|} y$ for all $y \in \underline{\boldsymbol{s}(i)}$;

3. If $\boldsymbol{t}\vec{m} \preceq \boldsymbol{s}$, then no element of $\vec{m}$ is in the set $\mathscr{A}_\alpha(\boldsymbol{t}) := \bigcup_{i=1}^{|\boldsymbol{t}|} \mathscr{A}_\epsilon(\boldsymbol{t})_\alpha(\boldsymbol{t}(i))$ whose elements are said to be ***unavailable*** at $\boldsymbol{t}$.

Notation. We write $\mathscr{J}_{\mathscr{A}}$ for the set of all justified sequences in a combinatorial arena $\mathscr{A}$.

The second axiom is a plain modification of the axiom on justified sequences in an arena that there is an edge between a non-initial element and its justifier [HO00]. The other two axioms take into account the *dynamics* of combinatorial arenas. The first axiom allows not only moves but also extended ones generated by exponential dynamics to be elements of a justified sequence $\boldsymbol{s}$ in a combinatorial arena, but only those with the *shortest* exponential tags. This restriction is for *finite presentations* of strategies (Definition 2.48); see the remark right after Definition 2.49. The third axiom requires that each element of $\boldsymbol{s}$ consists of *available* vertices only.

Example 2.20. Recall the combinatorial arena $\mathscr{A}$ given in Examples 2.8 and 2.17. The pairs

$$(yt.w.ys, \{1 \mapsto 0, 2 \mapsto 1, 3 \mapsto 0\}) \qquad (yt.ys.w, \{1 \mapsto 0, 2 \mapsto 0, 3 \mapsto 1\})$$

are both justified sequences in $\mathscr{A}$, but neither of the pairs

$$(yt.w.yz, \{1 \mapsto 0, 2 \mapsto 1, 3 \mapsto 0\}) \qquad (yt.yz.w, \{1 \mapsto 0, 2 \mapsto 0, 3 \mapsto 1\}).$$

This illustrates additive dynamics, which distinguishes with from tensor (cf. Example 2.12).

Next, as an illustration of exponential dynamics, which models the duplication of formulae given by of-course, observe that the pairs

$$(yx.w.y(z,x^1).y(x,x^2), \{1 \mapsto 0, 2 \mapsto 1, 3 \mapsto 0, 4 \mapsto 0\})$$

$$(yx.uv.y(x,x^1).(u,x^1)(v,x^1), \{1 \mapsto 0, 2 \mapsto 1, 3 \mapsto 0, 4 \mapsto 3\})$$

are both justified sequences in $\mathscr{A}$, but neither of the pairs

$$(yx.w.yz.yx, \{1 \mapsto 0, 2 \mapsto 1, 3 \mapsto 0, 4 \mapsto 0\}) \qquad (yx.uv.yx.uv, \{1 \mapsto 0, 2 \mapsto 1, 3 \mapsto 0, 4 \mapsto 3\}).$$

Convention. Let $\boldsymbol{s} = (\boldsymbol{s}, J_{\boldsymbol{s}})$ be a justified sequence in a combinatorial arena $\mathscr{A}$.

- An ***occurrence*** of an element $\vec{m}$ in $\boldsymbol{s}$ is a pair $(\boldsymbol{s}(i), i)$ such that $\boldsymbol{s}(i) = \vec{m}$ and $i \in \overline{|\boldsymbol{s}|}$;
- The occurrence $(\boldsymbol{s}(J_{\boldsymbol{s}}(i)), J_{\boldsymbol{s}}(i))$ is called the ***justifier*** of the one $(\boldsymbol{s}(i), i)$ in $\boldsymbol{s}$, and equivalently $(\boldsymbol{s}(i), i)$ is said to be ***justified*** by $(\boldsymbol{s}(J_{\boldsymbol{s}}(i)), J_{\boldsymbol{s}}(i))$ in $\boldsymbol{s}$;
- An occurrence $(\boldsymbol{s}(i), i)$ in $\boldsymbol{s}$ is said to be ***initial*** if $J_{\boldsymbol{s}}(i) = 0$, and ***pseudo-initial*** if $J_{\boldsymbol{s}}(i) > 0$ with all elements of $\boldsymbol{s}(i)$ the children of a switching element of $\boldsymbol{s}(J_{\boldsymbol{s}}(i))$.

Henceforth, we are casual about the distinction between extended moves and their occurrences in a sequence. Besides, we usually keep the pointer $J_{\boldsymbol{s}}$ of a justified sequence $\boldsymbol{s} = (\boldsymbol{s}, J_{\boldsymbol{s}})$ implicit and abbreviate occurrences $(\boldsymbol{s}(i), i)$ as $\boldsymbol{s}(i)$. We even write $J_{\boldsymbol{s}}(\boldsymbol{s}(i)) = \boldsymbol{s}(j)$ if $J_{\boldsymbol{s}}(i) = j$.

As a typical example of the use of this convention, we call a justified sequence $\boldsymbol{t}$ a ***justified subsequence*** of a justified sequence $\boldsymbol{s}$ if $\boldsymbol{t}$ is a subsequence of $\boldsymbol{s}$, and $J_{\boldsymbol{t}}(\vec{n}) = \vec{m}$ if and only if there are elements $\vec{m}_1, \vec{m}_2, \dots, \vec{m}_k$ ($k \in \mathbb{N}$) of $\boldsymbol{s}$ yet deleted in $\boldsymbol{t}$ such that $J_{\boldsymbol{s}}(\vec{n}) = \vec{m}_1$, $J_{\boldsymbol{s}}(\vec{m}_1) = \vec{m}_2$, $\dots$, $J_{\boldsymbol{s}}(\vec{m}_{k-1}) = \vec{m}_k$ and $J_{\boldsymbol{s}}(\vec{m}_k) = \vec{m}$.

As the last preparation for positions, let us adapt a central concept in [HO00], called *views*, to combinatorial arenas. In this adaptation, we modify views in such a way that they *forget all but the last exponential tags* on extended moves in order for *innocent* strategies (Definition 2.34) to be *finitely presentable* in a suitable sense (§2.4). We also let views handle the new concept of pseudo-initial occurrences in the same way as initial ones so that Theorem 3.26 will hold.

Definition 2.21 (views [Coq95, HO00])**.** The ***P-view*** $\lceil \boldsymbol{s} \rceil$ and the ***O-view*** $\lfloor \boldsymbol{s} \rfloor$ of a justified sequence $\boldsymbol{s}$ in a combinatorial arena $\mathscr{A}$ are respectively the justified sequences in $\mathscr{A}$ (consisting of moves in $\mathscr{A}$, not extended ones) defined inductively by

1. $\lceil \boldsymbol{\epsilon} \rceil := \boldsymbol{\epsilon}$;
2. $\lceil \boldsymbol{s}\vec{m} \rceil := \lceil \boldsymbol{s} \rceil.\vec{m}^\sharp$ if $\vec{m}$ is a P-move;
3. $\lceil \boldsymbol{s}\vec{m} \rceil := \vec{m}^\sharp$ if $\vec{m}$ is initial or pseudo-initial;
4. $\lceil \boldsymbol{s}\vec{m}\boldsymbol{t}\vec{n} \rceil := \lceil \boldsymbol{s} \rceil.\vec{m}^\sharp.\vec{n}^\sharp$ if $\vec{n}$ is an extended O-move such that $\vec{m}$ justifies $\vec{n}$;
5. $\lfloor \boldsymbol{\epsilon} \rfloor := \boldsymbol{\epsilon}$;
6. $\lfloor \boldsymbol{s}\vec{m} \rfloor := \lfloor \boldsymbol{s} \rfloor.\vec{m}^\sharp$ if $\vec{m}$ is an O-move;
7. $\lfloor \boldsymbol{s}\vec{m}\boldsymbol{t}\vec{n} \rfloor := \lfloor \boldsymbol{s} \rfloor.\vec{m}^\sharp.\vec{n}^\sharp$ if $\vec{n}$ is an extended P-move such that $\vec{m}$ justifies $\vec{n}$.

Remark. The P-view $\lceil \boldsymbol{s} \rceil$ or the O-view $\lfloor \boldsymbol{s} \rfloor$ may not be a justified sequence because the justifier of $\vec{m}$ may be lost in the clause (2) or (6). However, this problem is insignificant because we shall focus on a class of justified sequences, called *positions*, in which this problem does not occur.

The idea is that, for a nonempty position $\boldsymbol{s}\vec{m}$ in a combinatorial arena $\mathscr{A}$ such that $\vec{m}$ is of odd (respectively, even) depth, the O-view $\lfloor \boldsymbol{s} \rfloor$ (respectively, the P-view $\lceil \boldsymbol{s} \rceil$) is the currently 'relevant part' of $\boldsymbol{s}$ for Opponent (respectively, Player) from the logical standpoint. I.e., for logic Opponent (respectively, Player) is concerned only with the last occurrence in $\boldsymbol{s}$, its justifier and that justifier's O- (respectively, P-) view, which recursively proceeds. P-views are defined as such since intuitively Opponent's aim is to *attack* a specific point in Player's argument, and Player's goal is to *defend* against the attack; i.e., every extended O-move in a position attacks specifically the prefix of the position ending with the justifier of the extended O-move, so the P-view of the position focuses on the last one and the P-view of the prefix (recursively). O-views are *P-views in the domain of linear implication* since Opponent and Player are switched in the domain.

We are now ready to define *positions* in a combinatorial arena:

Notation. Let $\boldsymbol{s}$ be a justified sequence in a combinatorial arena $\mathscr{A}$.

- If $\mathscr{A} = \neg\mathscr{A}'$ and $\boldsymbol{s} = \neg\boldsymbol{t}$, then $\boldsymbol{s}\upharpoonright_{\mathscr{A}'} := \boldsymbol{t} \in \mathscr{J}_{\mathscr{A}'}$ (up to tags);
- If $\mathscr{A} = \mathscr{B} \times \mathscr{C}$, where $\times$ is $\otimes$ or $\&$, then let $\boldsymbol{s}\upharpoonright_{\mathscr{B}} \in \mathscr{J}_{\mathscr{B}}$ and $\boldsymbol{s}\upharpoonright_{\mathscr{C}} \in \mathscr{J}_{\mathscr{C}}$ be the justified subsequences of $\boldsymbol{s}$ (up to tags) that consist of extended moves in $\mathscr{B}$ and $\mathscr{C}$, respectively.
- If $\mathscr{A} = !\mathscr{A}''$, then each extended vertex of $\mathscr{A}$ is written uniquely in the form
$$x_0\{x_1x_2\dots x_n; 1^n\}\{y_1y_2\dots y_m; j_1j_2\dots j_m\} \quad (n, m \in \mathbb{N}) \tag{1}$$
such that $x_i \in \mathscr{R}_{|\mathscr{A}''|} \cap \mathscr{R}_{\mu_{\mathscr{A}''}}$ for all $i = 0, 1, \dots, n$ and $j_k > 1$ for $k = 1, 2, \dots, m$. Let $\boldsymbol{s}\upharpoonright_{1,l}$ be the justified subsequence of $\boldsymbol{s}$ consisting of extended moves in $\mathscr{A}$ whose elements are of the form (1) with $l = n$ except they are changed into
$$x_0\{y_1y_2\dots y_m; (j_1-1)(j_2-1)\dots(j_m-1)\}. \tag{2}$$
Note that $\boldsymbol{s}\upharpoonright_{1,l} \in \mathscr{J}_{\mathscr{A}''}$ holds thanks to the transformation of (1) into (2).
- If $\mathscr{A} = \mathscr{B} \multimap \mathscr{C}$, then $\boldsymbol{s}\upharpoonright_{\mathscr{B}} \in \mathscr{J}_{\mathscr{B}}$ is defined as above, and $\boldsymbol{s}\upharpoonright_{\mathscr{C}} \in \mathscr{J}_{\mathscr{C}}$ is obtained from the justified subsequence of $\boldsymbol{s}$ (up to tags) that consists of extended moves in $\mathscr{C}$ by deleting the switching (extended) vertex attributed to the linear implication.

Definition 2.22 (positions in a combinatorial arena)**.** A justified sequence $\boldsymbol{s}$ in a combinatorial arena $\mathscr{A}$ is called a ***position*** in $\mathscr{A}$ if it satisfies

1. (Recursive alternation) If $\boldsymbol{s} = \boldsymbol{t}\vec{m}\vec{n}\boldsymbol{u}$ with $\vec{m}$ an extended O- (respectively, P-) move, then $\vec{n}$ is an extended P- (respectively, O-) move, and this ***alternation*** between Opponent and Player holds recursively for $\boldsymbol{s}\upharpoonright_{\mathscr{A}'}$ in $\mathscr{A}'$ (respectively, for $\boldsymbol{s}\upharpoonright_{\mathscr{B}}$ in $\mathscr{B}$ and $\boldsymbol{s}\upharpoonright_{\mathscr{C}}$ in $\mathscr{C}$, for $\boldsymbol{s}\upharpoonright_{1,l}$ in $\mathscr{A}''$ for each $l \in \mathbb{N}$, for $\boldsymbol{s}\upharpoonright_{\mathscr{D}}$ in $\mathscr{D}$ and $\boldsymbol{s}\upharpoonright_{\mathscr{E}}$ in $\mathscr{E}$) if $\mathscr{A} = \neg\mathscr{A}'$ (respectively, if $\mathscr{A} = \mathscr{B} \times \mathscr{C}$, if $\mathscr{A} = !\mathscr{A}''$, if $\mathscr{A} = \mathscr{D} \multimap \mathscr{E}$);

2. (Visibility) If $\boldsymbol{s} = \boldsymbol{t}\vec{m}\boldsymbol{u}\vec{n}\boldsymbol{v}$ with $\vec{m}$ justifying $\vec{n}$, then $\vec{m}^\sharp$ occurs in the P-view $\lceil \boldsymbol{t}\vec{m}\boldsymbol{u} \rceil$ (respectively, in the O-view $\lfloor \boldsymbol{t}\vec{m}\boldsymbol{u} \rfloor$) when $\vec{n}$ is an extended P- (respectively, O-) move;

3. (Joker) Each element $\boldsymbol{s}(i)$ $(i \in \overline{|\boldsymbol{s}|})$ contains no extended vertex whose core is in $\alpha_{\mathscr{A}}[0]$.

Notation. We write $\mathcal{P}_{\mathscr{A}}$ for the set of all positions in a combinatorial arena $\mathscr{A}$.

Convention. A ***play*** in a combinatorial arena $\mathscr{A}$ refers to a nonempty, prefix-closed sequence of positions in $\mathscr{A}$. A play can be infinite in length; however, as in the case of other infinitary structures, we talk about infinite plays only temporarily; we will eventually dispense with them.

The recursive alternation axiom requires that every position is *alternating* not only globally but also locally and recursively. By Theorem 2.13, this condition is applied to *all* combinatorial arenas, not only to inductively constructed ones. Positions in an arena [HO00] only satisfy global alternation, which is insufficient to achieve fully complete semantics of linear logic. Meanwhile, fully complete game semantics of (fragments of) linear logic [AJ94, MO03, AM99c, Mel05] equips each game with a potentially infinite set of (selected) positions that satisfy recursive alternation. We shall obtain fully complete yet finitary semantics of linear logic partly by imposing recursive alternation in a finitary way (specifically via Lemma 3.22). Next, visibility [HO00] is a standard axiom in the literature; it ensures that each view is a justified sequence. Finally, the joker axiom is a novel one, and it is for our full completeness to admit 0-ary additives, i.e., one and zero.

Remark. An idea similar to the joker axiom, called *joker moves*, is used by Murawski and Ong [MO03] for their fully complete semantics of intuitionistic linear logic to include top. In contrast to our method, they interpret top, not one, by a joker move, and endow it with a more special status. Besides, the method of joker moves does not work in the presence of one or of-course.

Example 2.23. Among the justified sequences in the combinatorial arena $\mathscr{A}$ given in Example 2.20, only the first one is a position in $\mathscr{A}$; the other ones do not satisfy global alternation.

Next, the justified sequence $id.b.ic.a$ in the combinatorial arena $(\bot_{[a]} \otimes \bot_{[b]}) \multimap_{[i]} (\bot_{[c]} \otimes \bot_{[d]})$ is not a position since it does not satisfy recursive alternation (though it meets global alternation).

Further, the justified sequence $rg.mnf'.le'.e.f.e\{e';1\}$ in the combinatorial arena $((!\bot_{[e]} \multimap_{[l]} \bot_{[e']}) \multimap_{[m]} (\bot_{[f]} \multimap_{[n]} \bot_{[f']})) \multimap_{[r]} \bot_{[g]}$ is not a position as the last move violates visibility.

Finally, the justified sequence $io.p$ in the combinatorial arena $1_{[p]} \multimap_{[i]} 1_{[o]}$ is not a position because neither of the two moves satisfies joker.

Although positions in the examples given so far are not very eye-friendly, we can always make positions in a combinatorial arena readable by suitable *syntactic-sugars*. This is because each combinatorial arena has only finitely many vertices together with a finite number of vertex generators given by the exponential structure. The following example illustrates this point:

Example 2.24. The ***arithmetic combinatorial arena*** refers to the combinatorial arena $\mathscr{N} := !(\bot_{[q]} \multimap_{[p]} \bot_{[y]}) \otimes \bot_{[n]} \multimap_{[o]} \bot_{[\hat{q}]}$. Let us assign names to extended moves in $\mathscr{N}$ by

$$\vec{q}_0 := o\hat{q} \qquad \vec{\text{yes}}_i := py\{p^i;1^i\} \qquad \vec{q}_{i+1} := q\{p^i;1^i\} \qquad \vec{\text{no}} := n.$$

The arithmetic combinatorial arena $\mathscr{N}$ is a finitary recast of the *lazy natural number game* [Yam19, Example 23], and it defines natural numbers in a syntax-free, primitive, finitary fashion through the idea of a *counting game* as follows. First, each position in $\mathscr{N}$ is of the form

$$\vec{q}_0.\vec{\mathrm{yes}}_0.\vec{q}_1.\vec{\mathrm{yes}}_1 \dots \vec{q}_n.\vec{\mathrm{yes}}_n.\vec{q}_{n+1}.\vec{\mathrm{no}},$$

where $\vec{q}_0$ justifies $\vec{\mathrm{yes}}_i$ and $\vec{\mathrm{no}}$, and $\vec{\mathrm{yes}}_i$ justifies $\vec{q}_{i+1}$ $(0 \leqslant i \leqslant n)$. Then, this position can be read informally as follows:

1. Opponent asks 'Do you want to count one more?' by the initial move $\vec{q}_0$;
2. Player answers 'Yes!' by the move $\vec{\mathrm{yes}}_0$ if she does, and 'No!' by the move no otherwise, where the play stops in the latter case;
3. If the last move by Player is $\vec{\mathrm{yes}}_i$, then Opponent asks 'Do you want to count one more?' by the move $\vec{q}_{i+1}$;
4. Player answers 'Yes!' by the move $\vec{\mathrm{yes}}_{i+1}$ if she does, and 'No!' by the move $\vec{\mathrm{no}}$ otherwise, where again the play stops in the latter case;
5. Opponent and Player iterate the steps 3 and 4 until Player makes the move $\vec{\mathrm{no}}$.

2.3 Combinatorial sequents

Recall that games do not capture *intensionality* in logic and computation such as *cuts* [Gen36], but *dynamic games* [YA20], a generalisation of games, overcome this limitation. Based on that work, we generalise combinatorial arenas to *combinatorial sequents*, which may capture cuts.

Notation. Given a switching vertex v of a combinatorial arena $\mathscr{A}$, let $\mathscr{A}^v$ be the combinatorial subarena of $\mathscr{A}$ induced by moves containing v (i.e., $|\mathscr{A}^v| = |\mathscr{A}|_{\{\, u \in \mathscr{V}_{|\mathscr{A}|} \mid \exists m \in \mu_{\mathscr{A}} .\, \{u,v\} \subseteq \underline{m} \,\}}$).

Definition 2.25 (combinatorial sequents)**.** A ***combinatorial sequent*** $\mathscr{S}$ is a combinatorial arena $\underline{\mathscr{S}}$ together with a fraternal set $\mathrm{Int}_{\mathscr{S}} \subseteq \mathscr{V}^1_{|\underline{\mathscr{S}}|}$ of switching vertices such that

1. The unique parent of elements of $\mathrm{Int}_{\mathscr{S}}$ is also switching, and its children are all in $\mathrm{Int}_{\mathscr{S}}$;
2. For each $v \in \mathrm{Int}_{\mathscr{S}}$, $\underline{\mathscr{S}}^v \cong\, !^k(\mathscr{A} \multimap \mathscr{A})$ for some combinatorial arena $\mathscr{A}$ and number $k \in \mathbb{N}$;
3. If $v \in \mathrm{Int}_{\mathscr{S}}$ with $\{v, w\} \in \alpha_{\underline{\mathscr{S}}}[2]$ or $\{v, w\} \subseteq V \in \mathscr{V}_{\epsilon_{\underline{\mathscr{S}}}}$, then $w \in \mathrm{Int}_{\mathscr{S}}$,

where $\mathrm{Int}_{\mathscr{S}}$ is called the ***intensionality*** of $\mathscr{S}$. An element of $\alpha_{\underline{\mathscr{S}}}[2]$ or a vertex of $\mathscr{V}_{\epsilon_{\underline{\mathscr{S}}}}$ is said to be ***internal*** if its elements are all in $\mathrm{Int}_{\mathscr{S}}$, and ***external*** otherwise.

Example 2.26. The pair $\mathscr{S} = (\underline{\mathscr{S}}, \mathrm{Int}_{\mathscr{S}})$ of the combinatorial arena $\underline{\mathscr{S}} := \bot_{[a]} \multimap_{[b]} !((\bot_{[c]} \multimap_{[d]} !\bot_{[e]}) \multimap_{[f]} (\bot_{[g]} \multimap_{[h]} !\bot_{[i]}))\,\&\,(\top \multimap_{[j]} \top) \multimap_{[k]} \bot_{[l]}$ and the set $\mathrm{Int}_{\mathscr{S}} := \{f, j\}$ is a combinatorial sequent, where $\underline{\mathscr{S}}^f = !((\bot_{[c]} \multimap_{[d]} !\bot_{[e]}) \multimap_{[f]} (\bot_{[g]} \multimap_{[h]} !\bot_{[i]}))$ and $\underline{\mathscr{S}}^j = \top \multimap_{[j]} \top$.

The pair $((\bot_{[a]} \multimap_{[b]} \bot_{[c]}) \otimes (\top \multimap_{[d]} \top) \multimap_{[e]} \bot_{[f]}, \{b, d\})$ is a combinatorial sequent, but the similar pair $((\bot_{[a]} \multimap_{[b]} (\bot_{[c]} \otimes (\top \multimap_{[d]} \top))) \multimap_{[e]} \bot_{[f]}, \{b, d\})$ is not.

The name of combinatorial sequents comes from their free characterisation (Proposition 2.31), which resembles sequents [Gen36]. This characterisation exploits the axioms on combinatorial sequents. In fact, a combinatorial sequent $\mathscr{S}$ corresponds to a sequent in our variant of a sequent calculus (§3), where elements of $\mathrm{Int}_{\mathscr{S}}$ to cuts. Under this correspondence, the internal part of an additive or an exponential structure is between or on cuts in the sense clarified shortly.

Notation. Given a combinatorial sequent $\mathscr{S}$ together with an isomorphism $\underline{\mathscr{S}}^v \cong\ !^k(\mathscr{A} \multimap \mathscr{A})$ for some $v \in \mathrm{Int}_{\mathscr{S}}$, we write $\mathscr{S}(\vec{m}, \vec{n})$ or $\mathscr{S}(\vec{n}, \vec{m})$ if $\vec{m}$ is an extended move in one of the two copies of $!^k\mathscr{A}$, and $\vec{n}$ is the corresponding one in the other copy.

Definition 2.27 (positions in a combinatorial sequent)**.** A ***position*** in a combinatorial sequent $\mathscr{S}$ is a position $\boldsymbol{s}$ in $\underline{\mathscr{S}}$ such that if $\boldsymbol{t}\vec{m}\vec{n} \preceq \boldsymbol{s}$ is even, then $\mathscr{S}(\vec{m}, \vec{n})$.

Example 2.28. Recall the combinatorial sequent $\mathscr{S} = (\underline{\mathscr{S}}, \mathrm{Int}_{\mathscr{S}})$ defined in Example 2.26. The position $bkl.fhi.de.c.g.a$ in $\underline{\mathscr{S}}$ is a position in $\mathscr{S}$, but the one $bkl.fhi.de.c.a$ in $\underline{\mathscr{S}}$ is not.

That is, a position in a combinatorial sequent $\mathscr{S}$ is a position in $\underline{\mathscr{S}}$ in which each extended O-move $\vec{n}$ following an extended P-move $\vec{m}$ is the copy of $\vec{m}$ in the sense specified by $\mathrm{Int}(\mathscr{S})$. A combinatorial arena $\mathscr{A}$ is the same as the combinatorial sequent $(\mathscr{A}, \emptyset)$, and positions in $\mathscr{A}$ coincide with those in $(\mathscr{A}, \emptyset)$. Thus, combinatorial sequents generalise combinatorial arenas.

Notation. Given a combinatorial arena $\mathscr{A}$, we also write $\mathscr{A}$ for the combinatorial sequent $(\mathscr{A}, \emptyset)$.

Given a combinatorial sequent $\mathscr{S}$, we write $\mathcal{M}_{\mathscr{S}}$ (respectively, $\mathcal{M}^+_{\mathscr{S}}$, $\mathcal{P}_{\mathscr{S}}$) for the set of all moves (respectively, extended moves, positions) in the underlying combinatorial arena $\underline{\mathscr{S}}$.

Proposition 2.29 (closure of positions under views)**.** *If $\boldsymbol{s}$ is a position in a combinatorial sequent $\mathscr{S}$, then so are the P-view $\lceil \boldsymbol{s} \rceil$ and the O-view $\lfloor \boldsymbol{s} \rfloor$.*

Proof. By induction on the length $|\boldsymbol{s}|$ of $\boldsymbol{s}$. □

Finally, let us establish a free characterisation of combinatorial sequents:

Definition 2.30 (combinatorial cuts)**.** ***Combinatorial pre-cuts*** are the class of combinatorial arenas generated from linear implication of the form $\mathscr{A} \multimap \mathscr{A}$ with $\mathscr{A}$ a combinatorial arena by with and of-course, and ***combinatorial cuts*** are the class of combinatorial arenas generated from top and the same kind of linear implication by tensor, with and of-course.

Notation. Given combinatorial cuts $\mathscr{C}_j$ $(j \in \overline{m})$ and combinatorial arenas $\mathscr{A}_i$ $(i \in \overline{n})$ and $\mathscr{B}$, we write $\mathscr{A}_1, \mathscr{A}_2, \dots, \mathscr{A}_n \dashv \mathscr{C}_1, \mathscr{C}_2, \dots, \mathscr{C}_m \vdash \mathscr{B}$ for the combinatorial sequent $\mathscr{S}$ defined by

$$\underline{\mathscr{S}} := \bigotimes_{i=1}^{n} \mathscr{A}_i \multimap_{[\dashv]} \bigotimes_{j=1}^{m} \mathscr{C}_j \multimap_{[\vdash]} \mathscr{B} \qquad \mathrm{Int}_{\mathscr{S}} := \bigcup_{j=1}^{m} \{\, v \in \mathscr{R}_{|\mathscr{C}_j|} \mid v \text{ is switching in } \mu_{\mathscr{C}_j} \,\},$$

where $\bigotimes_{i=1}^{n} \mathscr{A}_i := \top$ if $n = 0$, $\bigotimes_{j=1}^{m} \mathscr{B}_j := \top$ if $m = 0$, and the identifiers $(\_)_{[\dashv]}$ and $(\_)_{[\vdash]}$ are fixed for convenience. We also write $\mathscr{A}_1, \mathscr{A}_2, \dots, \mathscr{A}_n \dashv \mathscr{C}_1, \mathscr{C}_2, \dots, \mathscr{C}_m \vdash$ for $\mathscr{S}$ if $\mathscr{B} = \bot$.

The letters Γ, Δ, etc. range over finite sequences of combinatorial arenas, and the ones Π, Σ, etc. over those of combinatorial cuts. Of-course is applied to these sequences componentwisely.

Proposition 2.31 (a free characterisation of combinatorial sequents)**.** *Every combinatorial sequent can be written, up to graph isomorphisms, in the form $\mathscr{A}_1, \mathscr{A}_2, \dots, \mathscr{A}_n \dashv \mathscr{C}_1, \mathscr{C}_2, \dots, \mathscr{C}_m \vdash \mathscr{B}$ for combinatorial cuts $\mathscr{C}_j$ $(j \in \overline{m})$ and combinatorial arenas $\mathscr{A}_i$ $(i \in \overline{n})$ and $\mathscr{B}$, and further made into the form $\mathscr{A}_1, \mathscr{A}_2, \dots, \mathscr{A}_n \dashv \mathscr{C} \vdash \mathscr{B}$ with $\mathscr{C}$ a combinatorial cut.*

Proof. The first part is by induction on the cardinality $|\mathrm{Int}_{\mathscr{S}}|$ for combinatorial sequents $\mathscr{S}$ with the help of the three axioms on $\mathscr{S}$ (Definition 2.25), and the second by replacing the case $m = 0$ with $\mathscr{C} := \top$, and the case $m > 0$ with $\mathscr{C} := \bigotimes_{j=1}^{m} \mathscr{C}_j$. □

This result is the second step towards the biequivalences between logic and combinatorics (§3.2). In addition, this free characterisation, together with Theorem 2.13, gives us a practical notation to concisely represent combinatorial sequents.

2.4 Finitely presentable strategies

Next, let us adapt another central concept in game semantics to our combinatorial setting:

Definition 2.32 (strategies [Nic94, AJM00, HO00]). A ***strategy*** on a combinatorial sequent $\mathscr{S}$ is a subset $\varphi \subseteq \mathcal{P}_{\mathscr{S}}^{\mathrm{Even}}$, written $\varphi : \mathscr{S}$, that is nonempty, *even-prefix-closed* (i.e., $\boldsymbol{s}\vec{m}\vec{n} \in \varphi$ implies $\boldsymbol{s} \in \varphi$) and *deterministic* (i.e., $\boldsymbol{s}\vec{m}\vec{n}, \boldsymbol{s}\vec{m}\vec{n}' \in \varphi$ implies $\boldsymbol{s}\vec{m}\vec{n} = \boldsymbol{s}\vec{m}\vec{n}'$).

A strategy $\varphi : \mathscr{S}$ depicts for Player *how to play* on the combinatorial sequent $\mathscr{S}$ by mapping an odd-length position $\boldsymbol{s}\vec{m}$ in $\mathscr{S}$ to its extension $\boldsymbol{s}\vec{m}\vec{n} \in \varphi$ if it exists (n.b., $\vec{m}$ is an extended *O-move*), where the extension, if any, is *unique* by the determinacy of φ, and the map is *partial* as there can be no extension of some odd-length position in φ. For convenience, we often confuse a strategy with the induced partial map from odd-length positions to extended P-moves.

Example 2.33. There are strategies $\{\boldsymbol{\epsilon}\}, \underline{n} : \mathscr{N}$ (Example 2.24), where $n \in \mathbb{N}$ and

$$\underline{n} := \{\, \boldsymbol{s} \in \mathcal{P}_{\mathscr{N}}^{\mathrm{Even}} \mid \boldsymbol{s} \preceq \vec{q}_0.\vec{\mathrm{yes}}_0.\vec{q}_1.\vec{\mathrm{yes}}_1 \ldots \vec{q}_n.\vec{\mathrm{yes}}_n.\vec{q}_{n+1}.\vec{\mathrm{no}} \,\}.$$

Loosely speaking, the trivial strategy $\{\boldsymbol{\epsilon}\}$ does nothing, while the one $\underline{n}$ plays by counting n.

In contrast to these finite strategies, the strategy on $!\neg\neg\top$ that computes by the unique map for each copy of $\neg\neg\top$ generated by of-course is *infinite*. Because our aim is to establish a *finitary* foundation (§1.1), we will eventually focus on *finitely presentable* strategies in a suitable sense.

Next, let us recall that not every strategy corresponds to a (formal) proof. For example, there is the trivial strategy on bottom $\bot := \neg\top$, but bottom models *falsity*, which has no proof. Thus, this strategy cannot be an interpretation of a proof. A standard way to carve out strategies for proofs is to impose *winning* [A$^+$97, §2] on strategies. We adapt this idea to our framework:

Definition 2.34 (winning strategies [HO00, CH10, Coq95, Lai97]). A strategy $\varphi : \mathscr{S}$ is

- ***Total*** if it always responds to the last extended O-move: If $\boldsymbol{s}\vec{m} \in \mathcal{P}_{\mathscr{S}}^{\mathrm{Odd}}$ and $\boldsymbol{s} \in \varphi$, then there is some (necessarily unique) $\boldsymbol{s}\vec{m}\vec{n} \in \varphi$;
- ***Innocent*** if it only depends on P-views: If $\boldsymbol{s}\vec{m}\vec{n} \in \varphi$, $\boldsymbol{t}\vec{l} \in \mathcal{P}_{\mathscr{S}}^{\mathrm{Odd}}$ and $\boldsymbol{t} \in \varphi$ satisfy $\lceil \boldsymbol{s}\vec{m} \rceil = \lceil \boldsymbol{t}\vec{l} \rceil$, then there is some $\boldsymbol{t}\vec{l}\vec{r} \in \varphi$ such that $\lceil \boldsymbol{s}\vec{m}\vec{n} \rceil = \lceil \boldsymbol{t}\vec{l}\vec{r} \rceil$;
- ***Noetherian*** if there is no strictly increasing (with respect to the prefix relation $\preceq$) infinite sequence of elements in the set $\lceil \varphi \rceil = \{\, \lceil \boldsymbol{s} \rceil \mid \boldsymbol{s} \in \varphi \,\}$ of all P-views in φ;
- ***Winning*** if it is total, innocent and noetherian.

Example 2.35. The trivial strategies $\{\boldsymbol{\epsilon}\} : \bot$ and $\{\boldsymbol{\epsilon}\} : \mathscr{N}$ are not total, while the ones $\{\boldsymbol{\epsilon}\} : \top$ and $\underline{n} : \mathscr{N}$ for all $n \in \mathbb{N}$ are winning.

One sees winning strategies as *proofs* as follows. First, proofs never get 'stuck,' so strategies for proofs must be *total*. Next, recall that views are the 'relevant' parts of positions from the logical perspective. In syntax, the 'irrelevant' parts correspond to *states*. Thus, imposing *innocence* on strategies corresponds to excluding *stateful* terms [AM97, AHM98, AM99a]. Logic is concerned with the *truth* of a formula, independently of 'states of arguments,' so strategies for proofs are innocent. Lastly, *noetherianity* is to decide who 'wins' in infinite plays: If a play by an innocent, noetherian strategy grows forever, then it is by Opponent so that the play is Player's 'win.'

In addition to this standard constraint on strategies in the literature, our full completeness result (Theorem 3.26) needs another one that corresponds to the *linearity* of proofs [Gir87]:

Definition 2.36 (linear strategies)**.** A P-move[3] $\vec{m}$ in a combinatorial sequent $\mathscr{S}$ is said to be

- ***Subject to O-joker*** if there is an edge $x \to y$ of $|\underline{\mathscr{S}}|$ such that $x \in \alpha_{\mathscr{S}}[0]$ and $y \in \underline{\vec{m}}$;
- ***Subject to P-exponential*** if an element of $\underline{\vec{m}}$ is a vertex of $|\underline{\mathscr{S}}|_V$ for some $V \in \mathscr{V}_{\epsilon_{\underline{\mathscr{S}}}}$.

An innocent strategy $\varphi : \mathscr{S}$ is said to be ***linear*** if the set $\{ \lceil \boldsymbol{s}\vec{o} \rceil \mid \boldsymbol{s}\vec{o}\vec{p} \in \varphi \}$ is singleton for each P-move $\vec{p} \in \mathcal{M}_{\mathscr{S}}$ not subject to O-joker or P-exponential, and ***linearly winning*** if it is linear and winning.

That is, a strategy is linear if it injectively visits all P-moves (in the *nondeterministic* sense explained in the examples below) in the underlying combinatorial arena except those in the effect of Opponent's one 1 or Player's of-course !. Intuitively, this axiom matches the linearity of proofs [Gir87]: A proof is linear precisely when it consumes and produces formulae (or resources) exactly as specified by the underlying sequent except those generated by the rule 1R or !W.

Remark. Murawski and Ong [MO03] impose *P-exhaustivity* on strategies, which is similar to our linearity, for their fully complete game semantics of the multiplicative fragment of intuitionistic linear logic. A difference between the two is that their approach entails a drastic modification of the very notion of (deterministic) strategies [MO03, p. 296], but ours does not.

Another existing game-semantic approach to linear proofs is *payoff functions* [MT10, §4]. In contrast to linearity or P-exhaustivity of strategies, payoff functions are extrinsic to the structure of games or strategies, and a priori they do not reflect the intuition behind linear proofs.

Example 2.37. The strategy on the combinatorial sequent $\bot_{[0]}, !\bot_{[1]}, \bot_{[2]} \dashv\vdash \bot_{[3]} \otimes \bot_{[4]}$ that maps $3 \mapsto 2$ and $4 \mapsto 0$ is linearly winning. The same computation also forms a linearly winning strategy on the combinatorial sequent $\bot_{[0]}, \bot_{[1]}, \bot_{[2]} \dashv\vdash \bot_{[3]} \otimes \bot_{[4]} \otimes 1_{[5]}$. In the both cases, the strategy injectively visits all P-moves not subject to O-joker or P-exponential *nondeterministically*: Any single play by the strategy does not cover all the P-moves, but its plays collectively do. The sequents $\bot_{[0]}, !\bot_{[1]}, \bot_{[2]} \vdash \bot_{[3]} \otimes \bot_{[4]}$ and $\bot_{[0]}, \bot_{[1]}, \bot_{[2]} \vdash \bot_{[3]} \otimes \bot_{[4]} \otimes 1_{[5]}$, which correspond to the two combinatorial sequents, respectively, are both provable in intuitionistic linear logic.

In contrast, the same computation yields a non-linear, though still winning, strategy on the combinatorial sequent $\bot_{[0]}, \bot_{[1]}, \bot_{[2]} \dashv\vdash \bot_{[3]} \otimes \bot_{[4]}$ due to the absence of of-course on $\bot_{[1]}$.

Another strategy on the first combinatorial sequent that plays $3 \mapsto 0$ and $4 \mapsto 0$ is winning but not linear: Its computation is (nondeterministically) not injective.

Example 2.38. The strategy on the combinatorial sequent $\bot_{[0]}, \bot_{[1]} \multimap \top, \bot_{[2]} \dashv\vdash \bot_{[3]}$ mapping $3 \mapsto 2$ and $1 \mapsto 0$ is linearly winning. The sequent $\bot_{[0]}, \bot_{[1]} \multimap \top, \bot_{[2]} \vdash \bot_{[3]}$, which corresponds to the combinatorial sequent, is provable in intuitionistic linear logic. This example illustrates one of the difficult challenges in achieving fully complete semantics of intuitionistic linear logic: Standard game semantics satisfies the equality $(\bot \multimap \top) = \top$, so it does not have a strategy that interprets a proof of the sequent. We solve this problem by our novel linear implication, which attains the inequality $(\bot \multimap \top) \neq \top$, and pseudo-initial occurrences; see Theorem 3.26.

Example 2.39. There are linearly winning strategies on the combinatorial sequent $\bot, \mathscr{A} \multimap \mathscr{A} \dashv\vdash \bot$ if $\mathscr{A}$ is top, bottom, $!\bot$, etc., but only winning, not linear, ones if $\mathscr{A}$ is 1 or $!(\bot \multimap \top)$. The corresponding sequent $\bot, \mathscr{A} \multimap \mathscr{A} \vdash \bot$ is provable in intuitionistic linear logic in the former cases, but not in the latter cases.

Let us proceed to a key algorithmic property of linearly winning strategies: *polynomial time decidability* (Theorem 2.44). This property is necessary for those strategies to constitute a formal system or *proof system* in the sense of [CR79]. To this end, we introduce the following concepts:

[3]Because linear strategies are innocent by definition, it suffices to focus on P-moves here, not extended ones.

Definition 2.40 (flatly innocent strategies)**.** A strategy $\varphi : \mathscr{S}$ is said to be ***flatly innocent*** if the relation $\mathrm{fun}_\varphi \subseteq \underline{\lceil \mathcal{P}^{\mathrm{Even}}_{\mathscr{S}} \rceil} \times \mathcal{M}_{\mathscr{S}} \times \mathcal{M}_{\mathscr{S}}$ given by $\mathrm{fun}_\varphi(\underline{\boldsymbol{s}}, \vec{o}, \vec{p}) :\Leftrightarrow \boldsymbol{s}\vec{o}\vec{p} \in \lceil \varphi \rceil$ forms a partial map $\underline{\lceil \mathcal{P}^{\mathrm{Even}}_{\mathscr{S}} \rceil} \times \mathcal{M}_{\mathscr{S}} \rightharpoonup \mathcal{M}_{\mathscr{S}}$ such that $\varphi = \tilde{\varphi}$ if $\mathrm{fun}_\varphi = \mathrm{fun}_{\tilde{\varphi}}$ for all innocent $\tilde{\varphi} : \mathscr{S}$.

While innocent strategies compute on P-views, flatly innocent ones on *flattened* P-views in the sense that they forget the order of elements of each input position except the last element.

Definition 2.41 (circular strategies)**.** An innocent strategy $\varphi : \mathscr{S}$ is said to be ***circular*** if $\mathrm{fun}_\varphi(S, \vec{o}, \vec{p})$ for some $S \in \underline{\lceil \mathcal{P}^{\mathrm{Even}}_{\mathscr{S}} \rceil}$ and $\vec{o}, \vec{p} \in \mathcal{M}_{\mathscr{S}}$ such that $\vec{p} \in S$ and $\vec{o}$ justifies $\vec{p}$.

In other words, an innocent strategy is circular if its computation exhibits *circularity* in the sense that it outputs what has been already played before.

Example 2.42. *Fixed-point strategies*, which will be given in Example 4.4, are an example of total innocent strategies that are not noetherian, flatly innocent or non-circular.

The following lemma is interesting in its own right since it implies that an innocent strategy fails to be noetherian precisely when it is circular or not flatly innocent. This characterisation holds thanks to the finitary nature of combinatorial arenas in the sense that an infinite growth of a P-view occurs only through the action of of-course yet the action is invisible in P-views.

Lemma 2.43 (flat innocence lemma)**.** *Every flatly innocent strategy is innocent, and an innocent strategy is flatly innocent if and only if it is non-circular if and only if it is noetherian.*

Proof. Clearly, a strategy is not flatly innocent if it it is not innocent; this verifies the first part.

For the second part, let $\varphi : \mathscr{S}$ be an innocent strategy. First, noetherianity clearly implies non-circularity. Next, let us show that flat innocence implies non-circularity. Assume that φ is circular. Then, fun_φ does not distinguish two different P-views (by flattening them), so φ is not flatly innocent. Finally, let us show that non-circularity implies noetherianity and flat innocence. Let φ be non-circular. Then, each P-view $\boldsymbol{s}\vec{m}\vec{n} \in \lceil \varphi \rceil$ does not have more than one occurrence of the same P-move. This also implies that the P-view does not admit more than one occurrence of the same O-move either because in a P-view each occurrence of a P-move justifies at most one occurrence. Hence, the triple $(\underline{\boldsymbol{s}}, \vec{m}, \vec{n})$ completely recovers the P-view $\boldsymbol{s}\vec{m}\vec{n}$. This verifies that φ is flatly innocent. The same argument also shows that φ is noetherian since otherwise one gets a contradiction to the finiteness of the set $\underline{\lceil \varphi \rceil}$. □

Because a strategy can be infinite, it is in general undecidable in conventional game semantics whether a given strategy is total, innocent or noetherian. Nevertheless, Lemma 2.43 implies that our combinatorial approach overcomes this undecidability in game semantics:

Theorem 2.44 (polynomial time decidability of winning)**.** *Let $\mathscr{S}$ be a combinatorial sequent. It only takes polynomial time to decide if a given (necessarily finite) relation $R \subseteq \underline{\lceil \mathcal{P}^{\mathrm{Even}}_{\mathscr{S}} \rceil} \times \mathcal{M}_{\mathscr{S}} \times \mathcal{M}_{\mathscr{S}}$ encodes a winning (respectively, linearly winning) strategy $\varphi : \mathscr{S}$ by $R = \mathrm{fun}_\varphi$.*

Proof. The following algorithm only takes polynomial time (with respect to the size of R):

1. Decide if R is a total map $\underline{\lceil \mathcal{P}^{\mathrm{Even}}_{\mathscr{S}} \rceil} \times \mathcal{M}_{\mathscr{S}} \to \mathcal{M}_{\mathscr{S}}$. Output 'No!' and stop if not; go to the next step otherwise.

2. Decide if R is circular. Output 'Yes!' and stop if not; output 'No!' and stop otherwise.

By Lemma 2.43, this algorithm outputs 'Yes!' (respectively, 'No!') if and only if $R = \mathrm{fun}_\varphi$ holds (respectively, does not hold) for a unique winning strategy $\varphi : \mathscr{S}$.

In addition, it is also polynomial time decidable whether a winning strategy is linear. □

Let us next turn to constructions on strategies. We adapt the novel, *intensional* constructions on strategies introduced by Yamada and Abramsky [YA20] to our combinatorial setting:

Notation. We generalise the notation $\upharpoonright$ used in Definition 2.22 and write $\boldsymbol{s}\upharpoonright_{\mathscr{A}_1,\mathscr{A}_2,\dots,\mathscr{A}_n}$ for the justified subsequence of a justified sequence $\boldsymbol{s}$ in a combinatorial arena $\mathscr{A}$, where $\mathscr{A}_i$ ($i \in \overline{n}$) are combinatorial subarenas of $\mathscr{A}$, that consists of extended moves in $\mathscr{A}_i$ for some $i \in \overline{n}$.

Definition 2.45 (constructions on strategies)**.** Given strategies $\varphi : (\Gamma \dashv \Pi \vdash \mathscr{A})$, $\varphi' : (\Gamma' \dashv \Pi' \vdash \mathscr{A}')$, $\psi : (\Gamma \dashv \Sigma \vdash \mathscr{B})$, $\tau : (\Delta, \mathscr{A} \dashv \Sigma \vdash \mathscr{B})$ and $\theta : (!\Gamma \dashv !\Pi \vdash \mathscr{A})$, we define

- The ***copy-cat*** $\mathrm{cp}_{\mathscr{A}} : (\mathscr{A}_{[0]} \dashv\vdash \mathscr{A}_{[1]})$ on $\mathscr{A}$ by

$$\mathrm{cp}_{\mathscr{A}} := \{\, \boldsymbol{s} \in \mathcal{P}^{\mathrm{Even}}_{\mathscr{A}_{[0]} \dashv\vdash \mathscr{A}_{[1]}} \mid \forall \boldsymbol{t} \preceq \boldsymbol{s}.\, \mathrm{Even}(\boldsymbol{t}) \Rightarrow \boldsymbol{t}\upharpoonright_{\mathscr{A}_{[0]}} = \boldsymbol{t}\upharpoonright_{\mathscr{A}_{[1]}}, \mathrm{Init}_{\mathscr{A}_{[0]},\mathscr{A}_{[1]}}(\boldsymbol{s}) \,\},$$

  where the predicate $\mathrm{Init}_{\mathscr{A}_{[0]},\mathscr{A}_{[1]}}(\boldsymbol{s})$ means that each initial occurrence in $\boldsymbol{s}$ on $\mathscr{A}_{[0]}$ is justified by the last initial one on $\mathscr{A}_{[1]}$;

- The ***parallel composition*** $\varphi \between \tau : (\Gamma, \Delta \dashv \Pi, \mathscr{A}_{[0]} \multimap \mathscr{A}_{[1]}, \Sigma \vdash \mathscr{B})$ of φ and τ by

$$\varphi \between \tau := \{\, \boldsymbol{s} \in \mathcal{P}^{\mathrm{Even}}_{\Gamma,\Delta \dashv \Pi, \mathscr{A}_{[0]} \multimap \mathscr{A}_{[1]}, \Sigma \vdash \mathscr{B}} \mid \boldsymbol{s}\upharpoonright_{\Gamma,\Pi,\mathscr{A}_{[0]}} \in \varphi, \boldsymbol{s}\upharpoonright_{\Delta,\mathscr{A}_{[1]},\Sigma,\mathscr{B}} \in \tau \,\};$$

- The ***tensor*** $\varphi \otimes \varphi' : (\Gamma, \Gamma' \dashv \Pi, \Pi' \vdash \mathscr{A} \otimes \mathscr{A}')$ of φ and φ' by

$$\varphi \otimes \varphi' := \{\, \boldsymbol{s} \in \mathcal{P}^{\mathrm{Even}}_{\Gamma,\Gamma' \dashv \Pi,\Pi' \vdash \mathscr{A} \otimes \mathscr{A}'} \mid \boldsymbol{s}\upharpoonright_{\Gamma,\Pi,\mathscr{A}} \in \varphi, \boldsymbol{s}\upharpoonright_{\Gamma',\Pi',\mathscr{A}'} \in \varphi' \,\};$$

- The ***pairing*** $\langle \varphi, \psi \rangle : (\Gamma \dashv \Pi \,\&\, \Sigma \vdash \mathscr{A} \,\&\, \mathscr{B})$ of φ and ψ, if Π and Σ are both singleton (n.b., this does not lose generality since Π and Σ are combinatorial cuts, not pre-cuts), by

$$\langle \varphi, \psi \rangle := \{\, \boldsymbol{s} \in \mathcal{P}^{\mathrm{Even}}_{\Gamma \dashv \Pi \& \Sigma \vdash \mathscr{A} \& \mathscr{B}} \mid \boldsymbol{s}\upharpoonright_{\Gamma,\Pi,\mathscr{A}} \in \varphi, \boldsymbol{s}\upharpoonright_{\Gamma,\Sigma,\mathscr{B}} \in \psi \,\};$$

- The ***dereliction*** $\mathrm{der}_{\mathscr{A}} : !\mathscr{A} \dashv\vdash \mathscr{A}$ on $\mathscr{A}$ by

$$\mathrm{der}_{\mathscr{A}} := \{\, \boldsymbol{s} \in \mathcal{P}^{\mathrm{Even}}_{!\mathscr{A} \dashv\vdash \mathscr{A}} \mid \forall \boldsymbol{t} \preceq \boldsymbol{s}.\, \mathrm{Even}(\boldsymbol{t}) \Rightarrow \boldsymbol{t}\upharpoonright_{!\mathscr{A}} = \boldsymbol{t}\upharpoonright_{\mathscr{A}}, \mathrm{Init}_{!\mathscr{A},\mathscr{A}}(\boldsymbol{s}) \,\};$$

- The ***promotion*** $\theta^\dagger : !\Gamma \dashv !\Pi \vdash !\mathscr{A}$ of θ by

$$\theta^\dagger := \{\, \boldsymbol{s} \in \mathcal{P}^{\mathrm{Even}}_{!\Gamma \dashv !\Pi \vdash !\mathscr{A}} \mid \boldsymbol{s}\upharpoonright_{!\Gamma,!\Pi,!\mathscr{A}} \in \theta \,\};$$

- The ***transpose*** $\lambda(\tau) : \Delta \dashv \Sigma \vdash \mathscr{A} \multimap \mathscr{B}$ of τ by adjusting the tags on $\mathscr{A}$ in τ.

There is one more operation imported from dynamic game semantics [YA20], which is significantly simplified thanks to the intensionality *explicitly* displayed in a combinatorial sequent:

Definition 2.46 (the big-step hiding operation)**.** The ***big-step hiding operation*** is the function $\mathcal{H}^\omega$ that maps each combinatorial strategy $\varphi : (\Gamma \dashv \Pi \vdash \Phi)$ to the one

$$\mathcal{H}^\omega(\varphi) := \{\, \boldsymbol{s}\upharpoonright_{\Gamma,\Phi} \mid \boldsymbol{s} \in \varphi \,\} : \mathcal{H}^\omega(\mathscr{S}) := (\Gamma \dashv\vdash \Phi),$$

and the ***hiding equivalence*** is the equivalence relation $\simeq^\omega_{\mathcal{H}}$ between strategies defined by

$$\varphi : \mathscr{S} \simeq^\omega_{\mathcal{H}} \varphi' : \mathscr{S}' :\Leftrightarrow \mathcal{H}^\omega(\varphi) = \mathcal{H}^\omega(\varphi') : \mathcal{H}^\omega(\mathscr{S}) = \mathcal{H}^\omega(\mathscr{S}').$$

The big-step hiding operation $\mathcal{H}^\omega$ deletes the intensionality of a combinatorial sequent and a strategy *in one go*. We refine it into a *step-by-step one* $\mathcal{H}$ by Corollary 3.29, whose countable iteration coincides with $\mathcal{H}^\omega$, in such a way that $\mathcal{H}$ corresponds precisely to cut-elimination.

Proposition 2.47 (well-defined constructions on strategies)**.** *Copy-cats and derelictions are linearly winning strategies; strategies are closed under parallel composition, tensor, pairing, promotion, transpose and hiding, and they preserve winning, linear winning and the hiding equivalence.*

Proof. Straightforward and left to the reader. □

At the end of the present section, let us introduce a *finite presentation* of a strategy:

Definition 2.48 (P-view algorithms)**.** A ***P-view algorithm*** on a combinatorial sequent $\mathscr{S}$ is a (necessarily finite) partial map $\underline{\lceil \mathcal{P}^{\mathrm{Even}}_{\mathscr{S}} \rceil} \times \mathcal{M}_{\mathscr{S}} \rightharpoonup \mathcal{M}_{\mathscr{S}}$.

Definition 2.49 (finitely presentable strategies)**.** The strategy $\mathrm{st}_{\mathscr{S}}(f) : \mathscr{S}$ ***finitely presented*** by a P-view algorithm f on a combinatorial sequent $\mathscr{S}$ is defined inductively by

$$\mathrm{st}_{\mathscr{S}}(f) := \{\boldsymbol{\epsilon}\} \cup \{\, \boldsymbol{s}\vec{m}\vec{n} \in \mathcal{P}^{\mathrm{Even}}_{\mathscr{S}} \mid \boldsymbol{s} \in \mathrm{st}_{\mathscr{S}}(f), f(\underline{\lceil \boldsymbol{s} \rceil}, \vec{m}^\sharp) = \vec{n}^\sharp \,\}. \tag{3}$$

A strategy $\varphi : \mathscr{S}$ is ***finitely presentable*** if there is a P-view algorithm f on $\mathscr{S}$ that satisfies the equation $\mathrm{st}_{\mathscr{S}}(f) = \varphi$, where f is called a ***finite presentation*** of φ.

Remark. By the second half of the first axiom of Definition 2.19, p in the equation (3) is *unique*. It then follows that this equation (3) yields a well-defined flatly innocent strategy. Clearly, every flatly innocent strategy is finitely presentable, but not vice versa as we shall see shortly.

Notation. We write $f :: \mathscr{S}$ if f is a P-view algorithm of a combinatorial sequent $\mathscr{S}$, and $\mathrm{st}(f)$ for the strategy $\mathrm{st}_{\mathscr{S}}(f) : \mathscr{S}$ when the underlying combinatorial sequent $\mathscr{S}$ is evident.

As the naming indicates, P-view algorithms are to serve as *finite presentations* of strategies. A finite presentation of a strategy exhibits *computability* of the strategy in the evident sense. We show in §4 that finitely presentable strategies form a powerful model of higher-order computation that is not only Turing complete but also *PCF-complete*, i.e., at least as strong as PCF.

As indicated by the notion of flat innocence, a P-view algorithm may finitely present more than one innocent strategy; i.e., P-view algorithms are more *abstract* than innocent strategies (by flattening P-views). This abstract nature of P-view algorithms is notable since it stands in sharp contrast with existing approaches to computability which rely on symbolic methods with superfluous details for defining computability of more abstract, semantic objects. As a result, one does not have to care if a property of a strategy in terms of its finite presentation is invariant under the choice of the finite presentation; the proof of Theorem 2.44 is an example. This is a strong advantage as it is often nontrivial to prove the invariance of a property under the choice of representations. More generally, inessential details of the symbolic manipulations in theory of computation, e.g., Turing machines and lambda-calculi, keep mathematician away from the field, but the semantic, abstract nature of our method changes this unfortunate situation.

Parallel composition, tensor and pairing take the disjoint unions of underlying combinatorial sequents, and promotion and transpose essentially do not alter innocent strategies. Hence:

Proposition 2.50 (preservation of flat innocence and finite presentability)**.** *Copy-cats and derelictions are flatly innocent and therefore finitely presentable; parallel composition, tensor, pairing, promotion and transpose preserve flat innocence and finite presentability.*

On the other hand, the big-step hiding operation does not preserve the finite presentability, let alone the flat innocence, of a strategy as the following example demonstrates:

Example 2.51. The strategies $\underline{0}, \underline{1} : \mathscr{N}$ are both linearly winning and flatly innocent, but the one $\underline{n} : \mathscr{N}$ is neither for each $n > 1$. How shall we finitely present the strategy $\underline{n} : \mathscr{N}$ then?

To answer this question, observe that the strategy $\mathrm{succ} : (\mathscr{N} \dashv\vdash \mathscr{N}')$, where the superscript $(\_)'$ is a tag to distinguish the two copies of the arithmetic combinatorial arena $\mathscr{N}$, defined by $\mathrm{succ} := \{ \dashv\vdash (\vec{q}_0)'.\vec{\mathrm{yes}}'_0((\vec{q}_1)'.\vec{q}_0.\vec{\mathrm{yes}}_0.\vec{\mathrm{yes}}'_1)((\vec{q}_2)'.\vec{q}_1.\vec{\mathrm{yes}}_1.\vec{\mathrm{yes}}'_2) \dots ((\vec{q}_n)'.\vec{q}_{n-1}.\vec{\mathrm{yes}}_{n-1}.\vec{\mathrm{yes}}'_n)((\vec{q}_{n+1})'.\vec{q}_n.\vec{\mathrm{no}}.\vec{\mathrm{no}}') \mid n \in \mathbb{N} \}$ is linearly winning and flatly innocent; we leave the verification to the reader. Then, the strategy

$$\mathrm{succ}^n(\underline{0}) := \underline{0} \between \mathrm{succ} \between \mathrm{succ} \between \cdots \between \mathrm{succ} : (\dashv (\mathscr{N} \multimap \mathscr{N})^n \vdash \mathscr{N})$$

is linearly winning and flatly innocent by Propositions 2.47 and 2.50, and has $\mathcal{H}^\omega(\mathrm{succ}^n(\underline{0})) = \underline{n}$. Hence, although $\underline{n}$ itself is not finitely presentable, its *intensional refinement* $\mathrm{succ}^n(\underline{0})$ is.

The above example also illustrates the fine analysis of *computational complexity* given by our intensional approach. For instance, take $n := 100$. Recall that the strategy $\underline{n} : \mathscr{N}$ answers each question by Opponent, and the answer depends on the number of previous question answering. Hence, it is intuitively much easier for the strategy $\underline{n}$ to answer the *initial* question by saying 'counting one more' than the *99th* question because in the first case it is immediately obvious if the answer is the correct one but not in the second case. However, the strategy $\underline{n}$ counts the both answers *equally as single steps*; i.e., it cannot reflect the algorithmic difference between the two answers. Nevertheless, by shifting from strategies to finitely presentable ones, 'too big' computational steps such as answering the 99th question in $\mathscr{N}$ are banned, and we are led to the intensional one $\mathrm{succ}^n(\underline{0})$ that captures the algorithmic difference between the two answers.

In comparison with Turing machines [Tur37], the standard foundation of computational complexity theory, the significance of this framework is that it is applicable to computational complexity of *higher* types (§4). Note that *higher-order computational complexity theory* lacks a proper mathematical foundation, and the concept of higher-order computational complexity has stopped at type 2. In contrast, our game-semantic model of computation given in §4 captures computation at arbitrarily higher types, and it seems to be a promising solution to this problem. For lack of space, however, we leave it as future work to exploit this research direction.

2.5 Combinatorial bicategories of logic and computation

This section studies the algebras of combinatorial arenas and finitely presentable strategies in terms of Bènabou's *bicategories* [Bén67]. Whereas ordinary semantics of logic and computation [LS88] identifies proofs and programs with their *values*, which makes its structure (1-)categories, Yamada and Abramsky [YA20] showed that its extension to non-values entails a generalisation of the categorical structure into a bicategorical one. We follow this bicategorical account.

In terms of our framework, their idea is as follows. A strategy $\varphi : \mathscr{S}$ is said to be *normalised* or a *value* if $\mathrm{Int}_{\mathscr{S}} = \emptyset$, and the big-step hiding operation $\mathcal{H}^\omega$ maps non-values to their values. In the (conventional) category of games, morphisms are normalised strategies, and the composition corresponds to the operation $\mathcal{H}^\omega(\_ \between \_)$ in our approach. Thus, for admitting non-values, it seems necessary to replace this composition with parallel composition. However, parallel composition does not satisfy the unit law, e.g., $(\mathrm{id}_{\mathscr{N}} \between \mathrm{succ}) \neq \mathrm{succ}$, where identities are copy-cats. The idea of Yamada and Abramsky is then to relax the category into a bicategory, where 2-cells are the hiding equivalence, because copy-cats satisfy the bicategorical unit law, e.g., $(\mathrm{id}_{\mathscr{N}} \between \mathrm{succ}) \simeq^\omega_{\mathcal{H}} \mathrm{succ}$.

Why should one study the algebras of our combinatorics? One reason is clarity: The familiar categorical language will help one understand exotic combinatorial structures and relate them to other mathematical structures. Another reason is generality: The present categorical framework, once established, will be general enough to be applied to other semantics as well.

We focus on a class of bicategories or *E-categories* as in [YA20].[4] An E-category is a category

[4]E-categories are called β-categories in [YA20]. We change the name to avoid the conflict with β-reduction.

except that the equality between morphisms is replaced with an equivalence relation. Thus, the composition and identities in an E-category respect the equivalence relation, so does a functor between E-categories or an *E-functor*, and so does a natural transformation between E-functors or an *E-natural transformation*. If we regard the equivalence relation as 2-cells, then each E-category forms a bicategory in the evident way [YA20]. For brevity, we work on E-categories:

Convention. We say that an operation on morphisms is *up to an equivalence relation* $\simeq$ between morphisms if the operation preserves the relation $\simeq$, and similarly for a property of morphisms.

Definition 2.52 (combinatorial bicategories)**.** The E-category $\mathcal{G}$ consists of the following data:

- An object is a combinatorial arena;
- A morphism $\mathscr{A} \to \mathscr{B}$ is a pair (Π, φ) of a finite sequence Π of combinatorial cuts and a finitely presentable strategy φ on $\mathscr{A} \dashv \Pi \vdash \mathscr{B}$ whose computation on Π satisfies

    1. Each internal edge $e = \{x, y\} \in \alpha_\Pi[2]$ with x or y occurring in φ has a unique edge $e' = \{x', y'\} \in \alpha_{\Gamma \dashv\vdash \Phi}[2]$ such that, for all $\boldsymbol{s}\vec{o}\boldsymbol{t}\vec{p} \in \lceil \varphi \rceil$ with $\vec{p}$ justified by $\vec{o}$, $e' \cap \underline{\vec{o}} \neq \emptyset$ if and only if $e \cap \underline{\vec{p}} \neq \emptyset$;
    2. Each internal vertex $V \in \mathscr{V}_{\epsilon_\Pi}$ with at least one of its elements occurring in φ admits a unique vertex $V' \in \mathscr{V}_{\epsilon_{\Gamma \dashv\vdash \Phi}}$ such that, for all $\boldsymbol{s}\vec{o}\boldsymbol{t}\vec{p} \in \lceil \varphi \rceil$ with $\vec{p}$ justified by $\vec{o}$, $\vec{p}$ is of the form $p_0\{y'; j'\}$ with $y' \in V'$ if and only if $\vec{o}$ is of the form $o_0\{y; j\}$ with $y \in V$,

    where the morphism (Π, φ) is said to be ***normalised*** or a ***value*** if $\Pi = \boldsymbol{\epsilon}$;
- The composition of morphisms $(\Pi, \varphi) : \mathscr{A} \to \mathscr{B}$ and $(\Sigma, \psi) : \mathscr{B} \to \mathscr{C}$ is the pair
$$(\Pi, \varphi); (\Sigma, \psi) := ((\Pi, \mathscr{B} \multimap \mathscr{B}, \Sigma), \varphi \between \psi) : \mathscr{A} \to \mathscr{C};$$
- The identity $\mathrm{id}_{\mathscr{A}}$ is the pair $(\boldsymbol{\epsilon}, \mathrm{cp}_{\mathscr{A}}) : \mathscr{A} \to \mathscr{A}$;
- The equivalence relation $\simeq_{\mathscr{A},\mathscr{B}}$ between morphisms $(\Pi, \varphi), (\Pi', \varphi') : \mathscr{A} \rightrightarrows \mathscr{B}$ is given by
$$(\Pi, \varphi) \simeq_{\mathscr{A},\mathscr{B}} (\Pi', \varphi') :\Leftrightarrow \mathcal{H}^\omega(\varphi) = \mathcal{H}^\omega(\varphi'),$$
i.e., $\simeq_{\mathscr{A},\mathscr{B}}$ is the hiding equivalence $\simeq^\omega_{\mathcal{H}}$, where we often omit the subscripts $(\_)_{\mathscr{A},\mathscr{B}}$ on $\simeq$.

An subcategory $\mathcal{LG}$ of $\mathcal{G}$ up to $\simeq$ is obtained from $\mathcal{G}$ by requiring the finitely presentable strategy φ of each morphism $(\Pi, \varphi) : \mathscr{A} \to \mathscr{B}$ to be linearly winning.

Notation. We also write $\varphi : (\mathscr{A} \dashv \Pi \vdash \mathscr{B})$ for a 1-cell $(\Pi, \varphi) : \mathscr{A} \to \mathscr{B}$ in these bicategories.

It is straightforward to verify that $\mathcal{G}$ and $\mathcal{LG}$ form E-categories, thence bicategories; we leave it as an exercise. As we shall see, the linear winning condition on 1-cells suffices for $\mathcal{LG}$ to attain fully complete semantics of intuitionistic linear logic with respect to cut-free proofs; its additional axioms on 1-cells are to extend the full completeness to proofs with cuts (Theorem 3.26).

Also, these E-categories form standard categorical semantics of intuitionistic linear logic:

Definition 2.53 (new Seely categories [Bie95])**.** A ***new Seely category*** is a symmetric monoidal closed category $\mathcal{C} = (\mathcal{C}, \otimes, \top, \multimap)$ that has finite products $(1, \&)$ and is equipped with a comonad $!$ on $\mathcal{C}$ such that the canonical adjunction between the co-Kleisli category $\mathcal{C}_!$ and $\mathcal{C}$ is monoidal.

Proposition 2.54 (combinatorial new Seely categories)**.** *The E-categories $\mathcal{G}$ and $\mathcal{LG}$ give rise to new Seely categories up to the hiding equivalence $\simeq$ between morphisms.*

Proof. The notations for the categorical constructions in Definition 2.53 signify the corresponding ones in $\mathcal{G}$ and $\mathcal{LG}$. We leave out the details because the 1-categorical case for conventional game semantics is well-known and outlined in [Hyl97]. $\square$

In conventional game semantics, the unit $\top$ and the terminal object 1 *coincide*. Clearly, this degeneracy makes it hopeless for the game semantics to achieve fully completeness for intuitionistic linear logic because the logic distinguishes top and one, the corresponding formulae.

In contrast, the E-category $\mathcal{LG}$ satisfies $\top \neq 1$. It is then interesting to see how top fails to be the unit for product in $\mathcal{LG}$. Assume for a contradiction that top is the unit for product in $\mathcal{LG}$. Then, there is a unique morphism $\tau : \bot \to \top$ because the assumption induces the isomorphism $\top \cong 1$. However, there is no such τ in $\mathcal{LG}$ due to the linearity axiom on τ, a contradiction.

Last but not least, the (co-)Kleisli constructions on $\mathcal{G}$ or $\mathcal{LG}$ associated to the comonad ! and the monads $\neg\neg$ and $? := \neg!\neg$ are well-defined as they respect the hiding equivalence $\simeq$. These constructions yield *cartesian closed categories*, *star-autonomous categories* [Bar06] and *control categories* [Sel01] up to $\simeq$; they are respectively the categorical semantics of intuitionistic logic [LS88], classical linear logic [See87] and classical logic [Sel01]. Although the following fact is not strictly necessary for the present work, it says that our combinatorics induces these categories:

Corollary 2.55 ((co-)Kleisli extensions to non-linearity and classicality)**.** *The co-Kleisli categories $\mathcal{G}_!$ and $\mathcal{LG}_!$ are cartesian closed, the Kleisli ones $\mathcal{G}_{\neg\neg}$ and $\mathcal{LG}_{\neg\neg}$ are star-autonomous, and the Kleisli ones $\mathcal{G}_{!?}$ and $\mathcal{LG}_{!?}$ are control, up to the hiding equivalence $\simeq$ on morphisms.*

As advertised in §1.1, the *consistency* of our logical bicategories are immediate:

Proposition 2.56 (consistency)**.** *The bicategories $\mathcal{LG}$, $\mathcal{LG}_{\neg\neg}$, $\mathcal{LG}_!$ and $\mathcal{LG}_{!?}$ are all consistent in the sense that they have no 1-cells $\top \to \bot$ or $\top \to 0$.*

Proof. By the totality of 1-cells in these bicategories. $\square$

3 Combinatorial formal systems

This section presents a combinatorial reformulation of formal systems. Specifically, we establish *biequivalences* between $\mathcal{LG}$ and a bicategory ILL of intuitionistic linear logic, between $\mathcal{LG}_{\neg\neg}$ and that CLL of classical linear logic, between $\mathcal{LG}_!$ and that IL of intuitionistic logic, and between $\mathcal{LG}_{!?}$ and that CL of classical logic. Moreover, we define for each of the combinatorial bicategories a step-by-step operation on 1-cells that precisely corresponds to the cut-elimination in the syntax. These results, together with Theorem 2.44, justify our combinatorics as formal systems.

This result resolves the bottleneck of mathematical semantics (§1.1) completely by recasting cuts and cut-elimination syntax-freely and non-inductively. Even when one focuses on cut-free proofs, our result solves a problem open for thirty years: fully complete semantics of intuitionistic linear logic. In addition, the correspondence between proofs and strategies are remarkably tight: One may directly and non-inductively read off proofs as strategies, and vice versa.

Also, our method captures the relation between the logics in terms of categorical algebras. This method admits an *intuitive* reading too: The Kleisli extension $(\_)_?$ allows *unlimited hypothesis consumptions*, and the co-Kleisli one $(\_)_!$ does *unlimited reasoning do-overs.*

As the syntax of the logics, we adopt the *sigma-calculus* [Yam23], a novel term calculus for intuitionistic linear logic and its (co-)Kleisli extensions. This calculus has the tightest correspondence with $\mathcal{LG}$ as mentioned above. This is our primary motivation to use the calculus. Besides, whilst other term calculi for linear logic [Abr93, BBDPH93, BP96, Bie99] suffer from complex *commuting conversions*, the sigma-calculus does not. This abstract nature of the sigma-calculus makes the correspondences between the combinatorics and the syntax plain and direct.

We first review the syntax of the logics in terms of the sigma-calculus and interpret them by combinatorics in §3.1. We finally prove that the interpretations form biequivalences in §3.2.

3.1 Linear, intuitionistic and classical logics

Let us first recall the formal languages of (propositional) linear, intuitionistic and classical logics:

Definition 3.1 (formulae in classical linear logic [Gir87])**.** Formulae in classical linear logic are the formal expressions defined by the grammar (of the *Backus–Naur form* [Bac59])

$$A, B := \top \mid \bot \mid 1 \mid 0 \mid A \otimes B \mid A \parr B \mid A \,\&\, B \mid A \oplus B \mid A^{\perp} \mid {!A} \mid {?A},$$

where $A \multimap B := \neg A \parr B$, and we call $\top$ ***top***, $\bot$ ***bottom***, 1 ***one***, 0 ***zero***, $\otimes$ ***tensor***, $\parr$ ***par***, $\&$ ***with***, $\oplus$ ***plus***, $(\_)^{\perp}$ ***linear negation***, $!$ ***of-course***, $?$ ***why-not***, and $\multimap$ ***linear implication***.

Remark. We do not include propositional variables as (atomic) formulae for the following reason. In the literature, fully complete game semantics of (a fragment of) linear logic with propositional variables was achieved [AJ94], and we may certainly adapt that method. However, the adaptation significantly digresses our main contributions. For the same reason, other authors [Lau04, Cla21] also exclude propositional variables from their fully complete semantics of linear logic.

Definition 3.2 (formulae in intuitionistic linear logic [GL87])**.** Formulae in intuitionistic linear logic are the class of formulae in classical linear logic defined by the grammar

$$A, B := \top \mid 1 \mid A \otimes B \mid A \,\&\, B \mid A \multimap B \mid {!A} \mid \neg A,$$

where $\bot := \neg\top$, and $\neg$ is called ***tensorial negation*** (adopted from [MT10]).

Remark. We do not include zero or plus in intuitionistic linear logic as in [Mel09, §3.2], while some authors [GL87, Tro91, Abr93] do. The reason is that the *sequential* computation in the bicategory $\mathcal{LG}$, which interprets intuitionistic linear logic, does not have coproducts.

Definition 3.3 (formulae in classical and intuitionistic logics [Fre79, Hey30, TS00])**.** Formulae in classical and intuitionistic logics are both the formal expressions defined by the grammar

$$A, B := \mathrm{tt} \mid \mathrm{ff} \mid A \wedge B \mid A \vee B \mid A \Rightarrow B,$$

where $\sim\!A := A \Rightarrow \mathrm{ff}$, and we call tt ***truth***, ff ***falsity***, $\wedge$ ***conjunction***, $\vee$ ***disjunction***, $\Rightarrow$ ***implication***, and $\sim$ ***negation***.

Let us next review the provability of these logics in terms of the *sigma-calculus* [Yam23]:

Definition 3.4 (the sigma-calculus [Yam23])**.** The ***sigma-calculus*** consists of the following:

- (Types) A ***type*** is a formula in intuitionistic linear logic equipped with positive integers i in the form of the subscripts $(\_)_{[i]}$ on tensorial negation, called an ***identifier***, defined by

  $$S, T := \top \mid 1 \mid S \otimes T \mid S \,\&\, T \mid S \multimap T \mid {!S} \mid \neg_{[i]} S \quad (i \in \mathbb{N}_+)$$

  and $\bot_{[i]} := \neg_{[i]}\top$ (the identifier $(\_)_{[0]}$ is reserved for the *empty codomain* defined below).

  Notation. We write $\underline{S}$ for the formula obtained from a type S by deleting all identifiers. This operation extends to finite sequences of types in the componentwise fashion. We write $\imath$, $\jmath$, etc. for an assignment of identifiers that transforms a formula into a type; let $T(\imath)$ be a type T with such an assignment $\imath$ explicit, and $T\{\imath'/\imath\} := T(\imath')$. Abusing notation, let $T\{i'/i\}$ be the type obtained from T by renaming the identifier $(\_)_{[i]}$ with the one $(\_)_{[i']}$.

- (Particles) An ***atom*** is a formal expression $[V]o \mapsto p$ such that $o, p \in \mathbb{N}$ and V is a finite set of natural numbers. A ***particle***, written $\mathcal{P}$, $\mathcal{A}$, etc., is a finite set of atoms.

  Notation. From an atom $a = ([V]o \mapsto p)$ and a number $i \in \mathbb{N}_+$, we obtain the atom $a\{i/0\}$ by replacing 0 with i occurring in a. This operation extends to a particle $\mathcal{P}$ by $\mathcal{P}\{i/0\} := \{\, a\{i/0\} \mid a \in \mathcal{P} \,\}$, and to a finite sequence $\mathcal{P}_1\mathcal{P}_2 \dots \mathcal{P}_n$ of particles by $(\mathcal{P}_1\mathcal{P}_2 \dots \mathcal{P}_n)\{i/0\} := \mathcal{P}_1\{i/0\}\mathcal{P}_2\{i/0\} \dots \mathcal{P}_n\{i/0\}$. We also define another atom $a[+0i] := ([V \cup \{0, i\}]o \mapsto p)$, and similarly this operation extends to a particle and a finite sequence of particles.

- (Contexts) A ***context*** is a finite sequence $(\mathcal{P}_i : T_i)_{i \in \overline{n}}$ of pairs, written $\mathcal{P}_i : T_i$, of a particle $\mathcal{P}_i$ and a type T_i, where the context is written $\gamma : \Gamma$ if $\gamma = (\mathcal{P}_i)_{i \in \overline{n}}$ and $\Gamma = (T_i)_{i \in \overline{n}}$.

- (Cuts) A ***cut*** is a formal expression defined by

$$C, D := \top \mid \mathrm{Cut}(T \mid T') \mid C \otimes D \mid C \,\&\, D \mid !C,$$

  where T and T' are types that satisfy the equation $\underline{T} = \underline{T'}$. An ***intensionality*** is a finite sequence $(\mathcal{M}_i : C_i)_{i \in \overline{n}}$ of pairs, written $\mathcal{M}_i : C_i$, of a particle $\mathcal{M}_i$ and a cut C_i, where the intensionality is written $\pi : \Pi$ if $\pi = (\mathcal{M}_i)_{i \in \overline{n}}$ and $\Pi = (C_i)_{i \in \overline{n}}$.

- (Sigma-sequents) A ***sigma-sequent*** is a formal expression $\Gamma \dashv \Pi \vdash \Phi$ such that Γ, called the ***domain***, is a finite sequence of types, Π is that of cuts, and Φ, called the ***codomain***, is that of types of length at most 1. We identify the empty codomain with the symbol $\bot_{[0]}$.

- (Raw-terms) A ***raw-term*** is a formal expression $\gamma : \Gamma \dashv \pi : \Pi \vdash \phi : \Phi$, identified up to renaming of identifiers (similarly to the *variable convention* in the lambda-calculus [B$^+$84]), such that $\gamma : \Gamma$ is a context, $\pi : \Pi$ is an intensionality, and $\phi : \Phi$ is a context of length at most 1, and the numbers occurring as identifiers in the raw-term are pairwise distinct.

  Notation. If $t = (\gamma : \Gamma \dashv \pi : \Pi \vdash \phi : \Phi)$ is a raw-term, then $\mathrm{Sq}(t) := \Gamma \dashv \Pi \vdash \Phi$. Clearly, t is recovered from $\mathrm{Sq}(t)$ and the set $\mathrm{Atm}(t)$ of all atoms in t. The *union* $t \cup u$ of raw-terms t and u such that $\mathrm{Sq}(t) = \mathrm{Sq}(u)$ is the raw-term on $\mathrm{Sq}(t)$ corresponding to $\mathrm{Atm}(t) \cup \mathrm{Atm}(u)$. In raw-terms, $e_1, e_2, \dots, e_n$ denotes a set $\{e_i\}_{i \in \overline{n}}$, and $s_1; s_2; \dots; s_m$ a sequence $(s_j)_{j \in \overline{m}}$.

- (Sigma-terms) A ***sigma-term*** is a raw-term derivable by the rules displayed in Figure 1, where identifiers are renamed, if necessary, in such a way that they are always *fresh.*

  Notation. If t is a sigma-sequent on a sigma-sequent F, then we write $t :: F$.

Example 3.5. Figure 2 displays a derivation of a sigma-term.

It is immediate to see that the sigma-calculus reformulates intuitionistic linear logic:

Notation. Let $\wr x \int$ be a finite sequence ranging over the empty one $\boldsymbol{\epsilon}$ or the singleton one x.

Proposition 3.6 (a term calculus for intuitionistic linear logic [Yam23])**.** *A sequent* $\underline{\Gamma} \vdash \wr \underline{T} \int$ *is derivable in the sequent calculus for intuitionistic linear logic if and only if there is a sigma-term* $\gamma : \Gamma \dashv \pi : \Pi \vdash \wr \mathcal{B} : T \int$*, where the sequent is derivable without cut if and only if* Π *is empty.*

Let us next explain the intuition behind the sigma-calculus [Yam23]. This intuition was based on game semantics and originally informal, but it can be made precise by our combinatorics: A sigma-sequent represents a combinatorial sequent, and a sigma-term a P-view algorithm.

$$(\mathrm{XL})\ \frac{\gamma : \Gamma; \mathcal{P} : S; \mathcal{P}' : S'; \delta : \Delta \dashv \pi : \Pi \vdash \phi : \Phi}{\gamma : \Gamma; \mathcal{P}' : S'; \mathcal{P} : S; \delta : \Delta \dashv \pi : \Pi \vdash \phi : \Phi} \qquad (!\mathrm{W})\ \frac{\gamma : \Gamma \dashv \pi : \Pi \vdash \phi : \Phi}{\gamma : \Gamma; \emptyset : !S \dashv \pi : \Pi \vdash \phi : \Phi}$$

$$(!\mathrm{C})\ \frac{\gamma : \Gamma; \mathcal{P} : !S(\iota); \mathcal{P}' : !S(\iota') \dashv \pi : \Pi \vdash \phi : \Phi}{\gamma : \Gamma; \mathcal{P} \cup \mathcal{P}'\{\iota/\iota'\} : !S(\iota) \dashv \pi : \Pi \vdash \phi : \Phi} \qquad (!\mathrm{D})\ \frac{\gamma : \Gamma; \mathcal{P} : S \dashv \pi : \Pi \vdash \phi : \Phi}{\gamma : \Gamma; \mathcal{P} : !S \dashv \pi : \Pi \vdash \phi : \Phi}$$

$$(!\mathrm{R})\ \frac{\delta : !\Delta \dashv \pi : \Pi \vdash \mathcal{A} : S}{\delta : !\Delta \dashv \pi : !\Pi \vdash \mathcal{A} : !S} \qquad (\textsc{Cut})\ \frac{\gamma : \Gamma \dashv \pi : \Pi \vdash \mathcal{A} : S(\iota) \qquad \delta : \Delta; \mathcal{P} : S(\jmath) \dashv \sigma : \Sigma \vdash \phi : \Phi}{\gamma : \Gamma; \delta : \Delta \dashv \pi : \Pi; \mathcal{A} \cup \mathcal{P} : \mathrm{Cut}(S(\iota) \mid S(\jmath)); \sigma : \Sigma \vdash \phi : \Phi}$$

$$(\top\mathrm{L})\ \frac{\gamma : \Gamma \dashv \pi : \Pi \vdash \phi : \Phi}{\gamma : \Gamma; \emptyset : \top \dashv \pi : \Pi \vdash \phi : \Phi} \qquad (\top\mathrm{R})\ \frac{}{\dashv\vdash \emptyset : \top} \qquad (1\mathrm{R})\ \frac{}{\Gamma \dashv\vdash \emptyset : 1}$$

$$(\otimes\mathrm{L})\ \frac{\gamma : \Gamma; \mathcal{P} : S; \mathcal{Q} : T \dashv \pi : \Pi \vdash \phi : \Phi}{\gamma : \Gamma; \mathcal{P} \cup \mathcal{Q} : S \otimes T \dashv \pi : \Pi \vdash \phi : \Phi} \qquad (\otimes\mathrm{R})\ \frac{\gamma : \Gamma \dashv \pi : \Pi \vdash \mathcal{A} : S \qquad \delta : \Delta \dashv \sigma : \Sigma \vdash \mathcal{B} : T}{\gamma : \Gamma; \delta : \Delta \dashv \pi : \Pi; \sigma : \Sigma \vdash \mathcal{A} \cup \mathcal{B} : S \otimes T}$$

$$(\&\mathrm{L})\ \frac{\gamma : \Gamma; \mathcal{P} : S \dashv \pi : \Pi \vdash \phi : \Phi}{\gamma : \Gamma; \mathcal{P} : S \,\&\, T \dashv \pi : \Pi \vdash \phi : \Phi} \qquad (\&\mathrm{L})\ \frac{\gamma : \Gamma; \mathcal{Q} : T \dashv \pi : \Pi \vdash \phi : \Phi}{\gamma : \Gamma; \mathcal{Q} : S \,\&\, T \dashv \pi : \Pi \vdash \phi : \Phi}$$

$$(\&\mathrm{R})\ \frac{\gamma : \Gamma \dashv \pi : \Pi \vdash \mathcal{A} : S \qquad \gamma' : \Gamma \dashv \sigma : \Sigma \vdash \mathcal{B} : T}{\gamma \cup \gamma' : \Gamma \dashv \pi \cup \sigma : \Pi \,\&\, \Sigma \vdash \mathcal{A} \cup \mathcal{B} : S \,\&\, T}\ (|\Pi| = 1 = |\Sigma|)$$

$$(\neg\mathrm{L})\ \frac{\gamma : \Gamma \dashv \pi : \Pi \vdash \mathcal{A} : S}{\gamma[+0i] : \Gamma; \mathcal{A}[+0i] \cup \{[\emptyset]0 \mapsto i\} : \neg_{[i]}S \dashv \pi[+0i] : \Pi \vdash} \qquad (\neg\mathrm{R})\ \frac{\gamma : \Gamma; \mathcal{P} : S \dashv \pi : \Pi \vdash}{\gamma\{i/0\} : \Gamma \dashv \pi\{i/0\} : \Pi \vdash \mathcal{P}\{i/0\} : \neg_{[i]}S}$$

$$(\multimap\mathrm{L})\ \frac{\gamma : \Gamma \dashv \pi : \Pi \vdash \mathcal{A} : S \qquad \delta : \Delta; \mathcal{Q} : T \dashv \sigma : \Sigma \vdash \phi : \Phi}{\gamma : \Gamma; \delta : \Delta; \mathcal{A} \cup \mathcal{Q} : S \multimap T \dashv \pi : \Pi; \sigma : \Sigma \vdash \phi : \Phi} \qquad (\multimap\mathrm{R})\ \frac{\gamma : \Gamma; \mathcal{P} : S \dashv \pi : \Pi \vdash \mathcal{B} : T}{\gamma : \Gamma \dashv \pi : \Pi \vdash \mathcal{P} \cup \mathcal{B} : S \multimap T}$$

Figure 1: The typing rules of the sigma-calculus

Corollary 3.7 (a surjection between formulae and combinatorial arenas)**.** *The assignment of a combinatorial arena $[\![A]\!]_{\mathcal{CG}}$ to each formula A in intuitionistic linear logic defined by*

$$[\![\top]\!]_{\mathcal{CG}} := \top \qquad [\![1]\!]_{\mathcal{CG}} := 1 \qquad [\![A \otimes B]\!]_{\mathcal{CG}} := [\![A]\!]_{\mathcal{CG}} \otimes [\![B]\!]_{\mathcal{CG}} \qquad [\![A \,\&\, B]\!]_{\mathcal{CG}} := [\![A]\!]_{\mathcal{CG}} \,\&\, [\![B]\!]_{\mathcal{CG}}$$

$$[\![A \multimap B]\!]_{\mathcal{CG}} := [\![A]\!]_{\mathcal{CG}} \multimap [\![B]\!]_{\mathcal{CG}} \qquad [\![!A]\!]_{\mathcal{CG}} := ![\![A]\!]_{\mathcal{CG}} \qquad [\![\neg A]\!]_{\mathcal{CG}} := \neg[\![A]\!]_{\mathcal{CG}}$$

forms a surjection.

Proof. Immediate from Theorem 2.13. $\square$

Remark. This surjection $[\![\_]\!]_{\mathcal{CG}}$ fails to be injective: $[\![\top \otimes \top]\!]_{\mathcal{CG}} = \top = [\![\top]\!]_{\mathcal{CG}}$.

Definition 3.8 (sigma-sequents as combinatorial sequents)**.** The map $[\![\_]\!]_{\mathcal{CG}}$ of Corollary 3.7 extends to a type T by $[\![T]\!]_{\mathcal{CG}} := [\![\underline{T}]\!]_{\mathcal{CG}}$, to a cut C by $[\![\mathrm{Cut}(T \mid T')]\!]_{\mathcal{CG}} := [\![T]\!]_{\mathcal{CG}} \multimap [\![T']\!]_{\mathcal{CG}}$, and to a sigma-sequent $F = (S_1, S_2, \dots, S_n \dashv C_1, C_2, \dots, C_m \vdash T)$ by

$$[\![F]\!]_{\mathcal{CG}} := [\![S_1]\!]_{\mathcal{CG}}, [\![S_2]\!]_{\mathcal{CG}}, \dots, [\![S_n]\!]_{\mathcal{CG}} \dashv [\![C_1]\!]_{\mathcal{CG}}, [\![C_2]\!]_{\mathcal{CG}}, \dots, [\![C_m]\!]_{\mathcal{CG}} \vdash [\![T]\!]_{\mathcal{CG}},$$

where the number i of each identifier $(\_)_{[i]}$ in F is inherited to the corresponding vertex in $[\![F]\!]_{\mathcal{CG}}$, and called an ***O-numeral*** if it is contained by an O-move, and a ***P-numeral*** otherwise.

Definition 3.9 (raw-terms as P-view algorithms)**.** Given a raw-term t on a sigma-sequent F, the P-view algorithm $\mathrm{Alg}(t)$ on the combinatorial sequent $[\![F]\!]_{\mathcal{CG}}$ is defined by

$$\mathrm{Alg}(t) := \{\, (M_V^F, m_o^F, m_p^F) \mid [V]o \mapsto p \text{ is an atom occurring in } t \,\},$$

where m_i^F for each number i is the move in $[\![F]\!]_{\mathcal{CG}}$ that contains i, and $M_V^F := \{\, m_i^F \mid i \in V \,\}$.

$$\dfrac{(!\mathrm{R})\,\dfrac{(!\mathrm{D})\,\dfrac{(\neg\mathrm{R})\,\dfrac{(\mathrm{XL})\,\dfrac{(\neg\mathrm{L})\,\dfrac{(\top\mathrm{L})\,\dfrac{(\top\mathrm{R})\,\dfrac{}{\dashv\vdash \emptyset:\top}}{\emptyset:\top\dashv\vdash\emptyset:\top}}{\emptyset:\top;[\emptyset]0\mapsto p:\bot_{[p]}\dashv\vdash}}{[\emptyset]0\mapsto p:\bot_{[p]};\emptyset:\top\dashv\vdash}}{[\emptyset]o\mapsto p:\bot_{[p]}\dashv\vdash\emptyset:\bot_{[o]}}}{[\emptyset]o\mapsto p:!\bot_{[p]}\dashv\vdash\emptyset:\bot_{[o]}}}{[\emptyset]o\mapsto p:!\bot_{[p]}\dashv\vdash\emptyset:!\bot_{[o]}} \quad (!\mathrm{C})\,\dfrac{(\otimes\mathrm{R})\,\dfrac{(!\mathrm{D})\,\dfrac{(\neg\mathrm{R})\,\dfrac{(\mathrm{XL})\,\dfrac{(\neg\mathrm{L})\,\dfrac{(\top\mathrm{L})\,\dfrac{(\top\mathrm{R})\,\dfrac{}{\dashv\vdash \emptyset:\top}}{\emptyset:\top\dashv\vdash\emptyset:\top}}{\emptyset:\top;[\emptyset]0\mapsto q:\bot_{[q]}\dashv\vdash}}{[\emptyset]0\mapsto q:\bot_{[q]};\emptyset:\top\dashv\vdash}}{[\emptyset]i\mapsto q:\bot_{[q]}\dashv\vdash\emptyset:\bot_{[i]}}}{[\emptyset]i\mapsto q:!\bot_{[q]}\dashv\vdash\emptyset:\bot_{[i]}} \quad (!\mathrm{D})\,\dfrac{(\neg\mathrm{R})\,\dfrac{(\mathrm{XL})\,\dfrac{(\neg\mathrm{L})\,\dfrac{(\top\mathrm{L})\,\dfrac{(\top\mathrm{R})\,\dfrac{}{\dashv\vdash \emptyset:\top}}{\emptyset:\top\dashv\vdash\emptyset:\top}}{\emptyset:\top;[\emptyset]0\mapsto q':\bot_{[q']}\dashv\vdash}}{[\emptyset]0\mapsto q':\bot_{[q']};\emptyset:\top\dashv\vdash}}{[\emptyset]j\mapsto q':\bot_{[q']}\dashv\vdash\emptyset:\bot_{[j]}}}{[\emptyset]j\mapsto q':!\bot_{[q']}\dashv\vdash\emptyset:\bot_{[j]}}}{[\emptyset]i\mapsto q:!\bot_{[q]};[\emptyset]j\mapsto q':!\bot_{[q']}\dashv\vdash\emptyset:\bot_{[i]}\otimes\bot_{[j]}}}{[\emptyset]i\mapsto q,[\emptyset]j\mapsto q:!\bot_{[q]}\dashv\vdash\emptyset:\bot_{[i]}\otimes\bot_{[j]}}}{[\emptyset]o\mapsto p:!\bot_{[p]}\dashv\mathrm{Cut}(\emptyset:\bot_{[o]}\mid[\emptyset]i\mapsto q,[\emptyset]j\mapsto q:!\bot_{[q]})\vdash\emptyset:\bot_{[i]}\otimes\bot_{[j]}}$$

Figure 2: An example of a sigma-term

Proposition 3.10 (sigma-terms as finite presentations)**.** *For every sigma-term $t :: F$, the pair $([\![F]\!]_{\mathcal{LG}}, [\![t]\!]_{\mathcal{LG}})$, where $[\![t]\!]_{\mathcal{LG}} := \mathrm{st}(\mathrm{Alg}(t))$, is a 1-cell in the bicategory $\mathcal{LG}$.*

Proof. By induction on a derivation of a sigma-term. □

In this way, each sigma-term $t :: F$ is read off directly and non-inductively as a finite presentation of the 1-cell $[\![t]\!]_{\mathcal{LG}} : [\![F]\!]_{\mathcal{LG}}$, where the 1-cell is said to be ***definable*** by the sigma-calculus. By construction, this direct reading is *injective*

Proposition 3.11 (injectivity)**.** *For all sigma-terms $t, t' :: F$, $[\![t]\!]_{\mathcal{LG}} = [\![t']\!]_{\mathcal{LG}}$ implies $t = t'$.*

Let us next recall the *cut-elimination* for the sigma-calculus [Yam23], which deletes cuts in a sigma-term in a step-by-step fashion. As the name indicates, this procedure corresponds under Proposition 3.6 to the cut-elimination procedure for intuitionistic linear logic [Gen36, TS00].

Notation. Let $\mathcal{P}$ be a particle, ι an assignment of identifiers to a type, and t a raw-term.

- $\mathcal{P}\upharpoonright_\iota := \{\, a \in \mathcal{P} \mid \mathbb{N}(a) \cap \underline{\iota} \neq \emptyset \,\}$ and $\mathcal{P}\downharpoonright_\iota := \mathcal{P} \setminus \mathcal{P}\upharpoonright_\iota$, where $\mathbb{N}(a)$ and $\underline{\iota}$ are the sets of all numbers occurring in a and ι, respectively. These operations extend to t via $\mathrm{Atm}(t)$.
- A pair $\mathcal{M} : \mathrm{Cut}(T(\iota) \mid T(\jmath))$ of a particle $\mathcal{M}$ and a cut $\mathrm{Cut}(T(\iota) \mid T(\jmath))$, called a ***cut-pair***, is also written $\mathrm{Cut}(\mathcal{M}\upharpoonright_\iota : T(\iota) \mid \mathcal{M}\upharpoonright_\jmath : T(\jmath))$.
- Given finite sequences $\boldsymbol{c}$ and $\boldsymbol{c'}$ of cut-pairs with $\boldsymbol{c}$ occurring in t, we write $t[\boldsymbol{c}]$ for t with the *hole* $[\_]$ assigned to $\boldsymbol{c}$, and $t[\boldsymbol{c}']$ for what is obtained from $t[\boldsymbol{c}]$ by replacing $\boldsymbol{c}$ with $\boldsymbol{c}'$.
- We write $t\{\iota'/\iota\}$ for the raw-term obtained from t by replacing the assignment ι of identifiers to the one ι', and $t\{i'/i\}$ for $t\{\iota'/\iota\}$ if ι and ι are single assignments of i and i', respectively.
- For a cut-pair $\mathrm{Cut}(\mathcal{L} : T(\iota) \mid \mathcal{R} : T(\jmath))$, we write $\#\mathcal{R} = 1$ if, for each P-numeral p occurring in $\jmath$, $\mathcal{R}$ contains a unique atom of the form $[V]o \mapsto p$ yet no other atoms.

Definition 3.12 (cut-elimination [Yam23])**.** The ***cut-elimination*** for the sigma-calculus is the union $\rightarrow := \bigcup_{i=1}^{7} \rightarrow_i$ of the binary relations $\rightarrow_i$ on raw-terms listed in Figure 3, and the ***big-step cut-elimination*** $\rightarrow^\omega$ is the reflexive, transitive closure of $\rightarrow$.

$$
\begin{gathered}
t[\mathrm{Cut}(\mathcal{L} : S(\iota) \mid \emptyset : S(\tilde{\imath}))] \rightarrow_1 t[\boldsymbol{\epsilon}]{\downharpoonright_{\iota}} \\
t[\mathrm{Cut}(\mathcal{L} : \neg_{[i]} S(\iota) \mid \mathcal{R}, [J]o \mapsto j : \neg_{[j]} S(\jmath))] \rightarrow_2 t[\mathrm{Cut}(\mathcal{R} : S(\jmath) \mid \mathcal{L} : S(\iota))]\{o/i\} \\
t[\mathrm{Cut}(\mathcal{L} : S(\iota) \otimes T(\jmath) \mid \mathcal{R} : S(\tilde{\imath}) \otimes T(\tilde{\jmath}))] \rightarrow_3 t[\mathrm{Cut}(\mathcal{L}{\upharpoonright_{\iota}} : S(\iota) \mid \mathcal{R}{\upharpoonright_{\tilde{\imath}}} : S(\tilde{\imath})); \mathrm{Cut}(\mathcal{L}{\upharpoonright_{\jmath}} : T(\jmath) \mid \mathcal{R}{\upharpoonright_{\tilde{\jmath}}} : T(\tilde{\jmath}))] \\
t[\mathrm{Cut}(\mathcal{L} : S(\iota) \mathbin{\&} T(\jmath) \mid \mathcal{R} : S(\tilde{\imath}) \mathbin{\&} T(\tilde{\jmath}))] \rightarrow_4 t[\mathrm{Cut}(\mathcal{L}{\upharpoonright_{\iota}} : S(\iota) \mid \mathcal{R}{\upharpoonright_{\tilde{\imath}}} : S(\tilde{\imath})); \mathrm{Cut}(\mathcal{L}{\upharpoonright_{\jmath}} : T(\jmath) \mid \mathcal{R}{\upharpoonright_{\tilde{\jmath}}} : T(\tilde{\jmath}))] \\
t[\mathrm{Cut}(\mathcal{L} : S(\iota) \multimap T(\jmath) \mid \mathcal{R} : S(\tilde{\imath}) \multimap T(\tilde{\jmath}))] \rightarrow_5 t[\mathrm{Cut}(\mathcal{R}{\upharpoonright_{\tilde{\imath}}} : S(\tilde{\imath}) \mid \mathcal{L}{\upharpoonright_{\iota}} : S(\iota)); \mathrm{Cut}(\mathcal{L}{\upharpoonright_{\jmath}} : T(\jmath) \mid \mathcal{R}{\upharpoonright_{\tilde{\jmath}}} : T(\tilde{\jmath}))] \\
t[\mathrm{Cut}(\mathcal{L} : {!S}(\iota) \mid \mathcal{R} : {!S}(\jmath))] \rightarrow_6 t[\mathrm{Cut}(\mathcal{L} : S(\iota) \mid \mathcal{R} : S(\jmath))] \quad (\text{if } \#\mathcal{R} = 1) \\
t[\mathrm{Cut}(\mathcal{L} : {!S}(\iota) \mid \mathcal{R}, \mathcal{Q} : {!S}(\jmath))] \rightarrow_7 t[\mathrm{Cut}(\mathcal{L} : {!S}(\iota) \mid \mathcal{R} : {!S}(\jmath)); \mathrm{Cut}(\mathcal{L}\{\iota'/\iota\} : {!S}(\iota') \mid \mathcal{Q}\{\jmath'/\jmath\} : {!S}(\jmath'))] \cup t{\upharpoonright_{\iota,\jmath}}\{\iota', \jmath'/\iota, \jmath\} \\
(\text{if } \#\mathcal{R} = 1 \text{ and } \mathcal{Q} \neq \emptyset)
\end{gathered}
$$

Figure 3: Cut-elimination for the sigma-calculus

Example 3.13. The sigma-term of Example 3.5 computes by the cut-elimination steps

$$
\begin{gathered}
[\emptyset]o \mapsto p : {!\bot_{[p]}} \dashv \mathrm{Cut}(\emptyset : {!\bot_{[o]}} \mid [\emptyset]i \mapsto q, [\emptyset]j \mapsto q : {!\bot_{[q]}}) \vdash \emptyset : \bot_{[i]} \otimes \bot_{[j]} \\
\downarrow \\
[\emptyset]o \mapsto p, [\emptyset]o' \mapsto p : {!\bot_{[p]}} \dashv \mathrm{Cut}(\emptyset : \bot_{[o]} \mid [\emptyset]i \mapsto q : \bot_{[q]}); \mathrm{Cut}(\emptyset : \bot_{[o']} \mid [\emptyset]j \mapsto q' : \bot_{[q']}) \vdash \emptyset : \bot_{[i]} \otimes \bot_{[j]} \\
\downarrow_* \\
[\emptyset]i \mapsto p, [\emptyset]j \mapsto p : {!\bot_{[p]}} \dashv \mathrm{Cut}(\emptyset : \top \mid \emptyset : \top); \mathrm{Cut}(\emptyset : \top \mid \emptyset : \top) \vdash \emptyset : \bot_{[i]} \otimes \bot_{[j]} \\
\downarrow_* \\
[\emptyset]i \mapsto p, [\emptyset]j \mapsto p : {!\bot_{[p]}} \dashv\vdash \emptyset : \bot_{[i]} \otimes \bot_{[j]}.
\end{gathered}
$$

Yamada [Yam23] has proven basic properties of the cut-elimination such as *subject reduction*, *confluence* and *strong normalisation*. These properties collectively imply:

Theorem 3.14 (correctness of cut-elimination [Yam23])**.** *If t is a sigma-term on a sigma-sequent $\Gamma \dashv \Pi \vdash \Phi$, then there is a finite sequence $t = t_0 \rightarrow t_1 \rightarrow t_2 \rightarrow \cdots \rightarrow t_n$ of cut-elimination steps such that each t_i ($0 \leqslant i \leqslant n$) is a sigma-term with t_n on $\Gamma \dashv\vdash \Phi$.*

Let us next define a bicategory out of the sigma-calculus similarly to the one $\mathcal{LG}$:

Definition 3.15 (the sigma-calculus as a bicategory)**.** The E-category ILL is defined as follows:

- An object is a type;
- A morphism $S \rightarrow T$ is a pair (Π, t) of a finite sequence Π of cuts and a sigma-term t on $S \dashv \Pi \vdash T$, where the morphism (Π, t) is said to be ***normalised*** or a ***value*** if $\Pi = \boldsymbol{\epsilon}$;
- The composition of morphisms $(\Pi, t) : S \rightarrow T(\iota)$ and $(\Sigma, u) : T(\iota') \rightarrow U$ is the pair
$$(\Pi, t); (\Sigma, u) := ((\Pi, \mathrm{Cut}(T(\iota) \mid T(\iota')), \Sigma), t \cup u) : S \rightarrow U;$$
- The identity id_S is the pair $(\boldsymbol{\epsilon}, \mathrm{id}_S)$, where the sigma-term $\mathrm{id}_S :: (S(\jmath) \dashv\vdash S(\jmath'))$ is defined by induction on the type S in the standard way (i.e., the same as the proof that the sequent $A \vdash A$ is *derivable* for all formulae A in the sequent calculus for intuitionistic linear logic);
- The equivalence relation $\simeq_{S,T}$ between morphisms $(\Pi, t), (\Pi', t') : S \rightrightarrows T$ is given by
$$(\Pi, t) \simeq_{S,T} (\Pi', t') :\Leftrightarrow t \rightarrow^* t'' \text{ and } t' \rightarrow^* t'' \text{ for the same sigma-term } t'' \text{ on } S \dashv\vdash T,$$
where we often omit the subscripts $(\_)_{S,T}$ on $\simeq$ (n.b., $\rightarrow^* = \rightarrow^\omega$ thanks to Theorem 3.14).

It is straightforward, albeit tedious, to show that ILL forms a well-defined E-category. Further, it forms a new Seely category up to $\simeq$ similarly to $\mathcal{LG}$ though we will not use this result. We leave the details to the reader. We shall eventually prove that the two bicategories are *biequivalent* by:

Proposition 3.16 (a semantic functor)**.** *The maps $[\![\_]\!]_{\mathcal{LG}}$ give rise to a 2-functor (or E-functor) $\mathsf{ILL} \to \mathcal{LG}$, and it preserves the structures of new Seely categories up to 2-cells.*

Proof. Immediate from the constructions of ILL and $\mathcal{LG}$. $\square$

Next, Yamada [Yam23] employed the *double negation translation* of classical linear logic into intuitionistic linear logic [Tro91, §5.12] for the sigma-calculus to embody classical linear logic:

Proposition 3.17 (a term calculus for classical linear logic [Yam23])**.** *Let $\mathscr{T}_{\neg\neg}$ be the map from formulae in classical linear logic to those in intuitionistic linear logic defined by*

$$\mathscr{T}_{\neg\neg}(\top) := \top \qquad \mathscr{T}_{\neg\neg}(1) := 1 \qquad \mathscr{T}_{\neg\neg}(\bot) := \bot \qquad \mathscr{T}_{\neg\neg}(0) := \neg 1 \qquad \mathscr{T}_{\neg\neg}(A^{\bot}) := \neg \mathscr{T}_{\neg\neg}(A)$$

$$\mathscr{T}_{\neg\neg}(A \otimes B) := \neg\neg\mathscr{T}_{\neg\neg}(A) \otimes \neg\neg\mathscr{T}_{\neg\neg}(B) \qquad \mathscr{T}_{\neg\neg}(A \mathbin{\&} B) := \neg\neg\mathscr{T}_{\neg\neg}(A) \mathbin{\&} \neg\neg\mathscr{T}_{\neg\neg}(B)$$

$$\mathscr{T}_{\neg\neg}(A \parr B) := \neg(\neg\mathscr{T}_{\neg\neg}(A) \otimes \neg\mathscr{T}_{\neg\neg}(B)) \qquad \mathscr{T}_{\neg\neg}(A \oplus B) := \neg(\neg\mathscr{T}_{\neg\neg}(A) \mathbin{\&} \neg\mathscr{T}_{\neg\neg}(B))$$

$$\mathscr{T}_{\neg\neg}(A \multimap B) := \mathscr{T}_{\neg\neg}(A) \multimap \neg\neg\mathscr{T}_{\neg\neg}(B) \qquad \mathscr{T}_{\neg\neg}(!A) := !\neg\neg\mathscr{T}_{\neg\neg}(A) \qquad \mathscr{T}_{\neg\neg}(?A) := \neg!\neg\mathscr{T}_{\neg\neg}(A).$$

A sequent $\boldsymbol{A} \vdash \wr B \int$ is derivable in the sequent calculus for classical linear logic if and only if there is a sigma-term $\gamma : \Gamma \dashv \pi : \Pi \vdash \wr \mathcal{B} : T \int$ such that $\underline{\Gamma} = \mathscr{T}^{*}_{\neg\neg}(\boldsymbol{A})$ and $\underline{T} = \neg\neg\mathscr{T}_{\neg\neg}(B)$ if $\wr B \int = B$ and $\wr \mathcal{B} : T \int = \mathcal{B} : T$, where the sequent is derivable without cut if and only if $\Pi = \boldsymbol{\epsilon}$.

Moreover, the sigma-calculus also embodies intuitionistic logic through *Girard's translation* [GL87, §3.3] of intuitionistic logic into intuitionistic linear logic:

Proposition 3.18 (a term calculus for intuitionistic logic [Yam23])**.** *Let $\mathscr{T}_{!}$ be the map from formulae in intuitionistic logic to those in intuitionistic linear logic defined by*

$$\mathscr{T}_{!}(\mathrm{tt}) := \top \qquad \mathscr{T}_{!}(\mathrm{ff}) := \bot \qquad \mathscr{T}_{!}(A \wedge B) := \mathscr{T}_{!}(A) \mathbin{\&} \mathscr{T}_{!}(B)$$

$$\mathscr{T}_{!}(A \vee B) := \neg(\neg!\mathscr{T}_{!}(A) \mathbin{\&} \neg!\mathscr{T}_{!}(B)) \qquad \mathscr{T}_{!}(A \Rightarrow B) := !\mathscr{T}_{!}(A) \multimap \mathscr{T}_{!}(B).$$

A sequent $\boldsymbol{A} \vdash \wr B \int$ is derivable in the sequent calculus for intuitionistic logic if and only if there is a sigma-term $\gamma : \Gamma \dashv \pi : \Pi \vdash \wr \mathcal{B} : T \int$ such that $\underline{\Gamma} = !\mathscr{T}_{!}{}^{*}(\boldsymbol{A})$ and $\underline{\Phi} = \mathscr{T}_{!}(B)$ if $\wr B \int = B$ and $\wr \mathcal{B} : T \int = \mathcal{B} : T$, where the sequent is derivable without cut if and only if $\Pi = \boldsymbol{\epsilon}$.

Finally, the sigma-calculus also implements classical logic via *T-translation* [DJS95, DJS97]:

Proposition 3.19 (a term calculus for classical logic [Yam23])**.** *Let $\mathscr{T}_{!?}$ be the map from formulae in classical logic to those in intuitionistic linear logic defined by*

$$\mathscr{T}_{!?}(\mathrm{tt}) := \top \qquad \mathscr{T}_{!?}(\mathrm{ff}) := \bot \qquad \mathscr{T}_{!?}(A \wedge B) := ?\mathscr{T}_{!?}(A) \mathbin{\&} ?\mathscr{T}_{!?}(B)$$

$$\mathscr{T}_{!?}(A \vee B) := \neg(!\neg\mathscr{T}_{!?}(A) \mathbin{\&} !\neg\mathscr{T}_{!?}(B)) \qquad \mathscr{T}_{!?}(A \Rightarrow B) := !?\mathscr{T}_{!?}(A) \multimap ?\mathscr{T}_{!?}(B).$$

A sequent $\boldsymbol{A} \vdash [B]$ is derivable in the sequent calculus for classical logic if and only if there is a sigma-term $\gamma : \Gamma \dashv \pi : \Pi \vdash \wr \mathcal{B} : T \int$ such that $\underline{\Gamma} = !?\mathscr{T}_{!}{}^{*}(\boldsymbol{A})$ and $\underline{T} = ?\mathscr{T}_{!}(B)$ when $\wr B \int = B$ and $\wr \mathcal{B} : T \int = \mathcal{B} : T$, where the sequent is derivable without cut if and only if $\Pi = \boldsymbol{\epsilon}$.

Similarly to the case of the bicategory $\mathcal{LG}$, let us apply (co-)Kleisli constructions to ILL:

Definition 3.20 (syntactic (co-)Kleisli constructions)**.** The bicategories CLL for classical linear logic, IL for intuitionistic logic and CL for classical logic are the (co-)Kleisli constructions

$$\mathsf{CLL} := \mathsf{ILL}_{\neg\neg} \qquad\qquad \mathsf{IL} := \mathsf{ILL}_{!} \qquad\qquad \mathsf{CL} := \mathsf{ILL}_{!?}.$$

As expected, CLL forms a star-autonomous category, IL a cartesian closed category, and CL a control category, all up to $\simeq$. By the above results, these bicategories embody the respective logics, whose categorical structures correspond to logical constants and constructions. We show in the next section the *biequivalences* $\mathsf{ILL} \simeq \mathcal{LG}$, $\mathsf{CLL} \simeq \mathcal{LG}_{\neg\neg}$, $\mathsf{IL} \simeq \mathcal{LG}_{!}$ and $\mathsf{CL} \simeq \mathcal{LG}_{!?}$.

3.2 Biequivalences between formal systems and combinatorics

This section is to establish *biequivalences* between formal systems and combinatorics for linear, intuitionistic and classical logics, respectively, where *proofs may contain cuts.* Such *intensional* biequivalences are, to the best of our knowledge, achieved for the first time in the literature.

We begin with proving that the map $[\![\_]\!]_{\mathcal{LG}}$ from sigma-terms to 1-cells in the bicategory $\mathcal{LG}$ (Definition 3.9) is *surjective* (Theorem 3.26). To this end, we need two technical lemmata:

Definition 3.21 (the domain and the codomain of a combinatorial cut)**.** Given a combinatorial cut $\mathscr{C}$, its ***domain*** $\mathrm{dom}(\mathscr{C})$ and ***codomain*** $\mathrm{cod}(\mathscr{C})$ are the combinatorial arenas defined by

$$\mathrm{dom}(\mathscr{A}_{[0]} \multimap \mathscr{A}_{[1]}) := \mathscr{A}_{[0]} \qquad\qquad \mathrm{cod}(\mathscr{A}_{[0]} \multimap \mathscr{A}_{[1]}) := \mathscr{A}_{[1]}$$

$$\mathrm{dom}(\mathscr{C} \mathbin{\&} \mathscr{C}') := \mathrm{dom}(\mathscr{C}) \mathbin{\&} \mathrm{dom}(\mathscr{C}') \qquad\qquad \mathrm{cod}(\mathscr{C} \mathbin{\&} \mathscr{C}') := \mathrm{cod}(\mathscr{C}) \mathbin{\&} \mathrm{cod}(\mathscr{C}')$$

$$\mathrm{dom}(!\mathscr{C}) := !\mathrm{dom}(\mathscr{C}) \qquad\qquad \mathrm{cod}(!\mathscr{C}) := !\mathrm{cod}(\mathscr{C}) \qquad\qquad \mathrm{dom}(\top) := \mathrm{cod}(\top) := \top$$

$$\mathrm{dom}(\mathscr{C} \otimes \mathscr{C}') := \mathrm{dom}(\mathscr{C}) \otimes \mathrm{dom}(\mathscr{C}') \qquad\qquad \mathrm{cod}(\mathscr{C} \otimes \mathscr{C}') := \mathrm{cod}(\mathscr{C}) \otimes \mathrm{cod}(\mathscr{C}').$$

Lemma 3.22 (acyclicity lemma)**.** *Let $\varphi : [\![F]\!]_{\mathcal{LG}}$ be a 1-cell in the bicategory $\mathcal{LG}$ with F a sigma-sequent. If $<_\varphi$ is the relation on the set of all occurrences of linear implication in the domain of F and of cuts in the intensionality of F such that $X <_\varphi X'$ if and only if there is $\boldsymbol{s}\vec{m}\boldsymbol{t}\vec{n} \in \lceil \varphi \rceil$ with $\vec{m}$ in the domain of $[\![X']\!]_{\mathcal{LG}}$, and $\vec{n}$ in the codomain of $[\![X]\!]_{\mathcal{LG}}$ and justified by $\vec{m}$, then the relation constitutes the directed edges $X \to X'$ of a finite rooted dag.*

Proof. By the recursive alternation axiom on positions and Proposition 2.29, the directed edges never constitute a cycle. □

Definition 3.23 (the size of a sigma-sequent)**.** The ***size*** of a formula A in intuitionistic linear logic (Definition 3.2) is the positive integer $|A|$ defined by

$$|\top| := |1| := 1 \qquad |\neg A| := |!A| := |A| + 1 \qquad |A \otimes B| := |A \mathbin{\&} B| := |A \multimap B| := |A| + |B| + 1,$$

that $|T|$ of a type T is $|\underline{T}|$, that $|\mathrm{Cut}(T \mid T')|$ of a cut $\mathrm{Cut}(T \mid T')$ is $|T|$, and that $|\Gamma \dashv \Pi \vdash \Phi|$ of a sigma-sequent $\Gamma \dashv \Pi \vdash \Phi$ is $|\Gamma| + |\Pi| + |\Phi|$, where $|\Gamma|$ is the sum of the sizes of the components of Γ, and similarly for $|\Pi|$ and $|\Phi|$.

Lemma 3.24 (prime lemma)**.** *Given a sigma-sequent F and a 1-cell $\varphi : [\![F]\!]_{\mathcal{LG}}$ in the bicategory $\mathcal{LG}$, there is a finitely-indexed family $\{(F_i, \varphi_i)\}_{i \in I}$ of such pairs that satisfies*

1. *$|F_i| < |F|$ for each $i \in I$;*
2. *φ is definable by a sigma-term $t :: F$ if so is φ_i by a sigma-term $t_i :: F_i$ for each $i \in I$;*

3. *For each $i \in I$, the intensionality of F_i consists of of-course and/or with, the domain of one, negation, with and/or of-course, the codomain of top, one, tensor, with and/or of-course, and $\lceil \varphi_i \rceil$ visits an occurrence of of-course in the domain at most once.*

Proof. Let $F = \Gamma \dashv \Pi \vdash \Phi$. We first verify the claim on of-course in the third clause. Assume $\Gamma = \Delta, !S, \Theta$, and that $\lceil \varphi \rceil$ visits the occurrence $!S$ at most n-times. By taking sufficiently large n, we do not lose generality. Clearly, there is a 1-cell $\varphi' : [\![\Delta, (!S)^n, \Theta \dashv \Pi \vdash \Phi]\!]_{\mathcal{LG}}$ in $\mathcal{LG}$ that coincides with φ except that $\lceil \varphi' \rceil$ visits each component $!S$ of $(!S)^n$ at most once. We iterate this procedure until we obtain a sigma-sequent F_1 and a 1-cell $\varphi_1 : [\![F_1]\!]_{\mathcal{LG}}$ such that $\lceil \varphi_1 \rceil$ visits each occurrence of of-course in the domain at most once. If there is a sigma-term $t_1 :: F_1$ such that $[\![t_1]\!]_{\mathcal{LG}} = \varphi_1$, then by the rules !C and XL, we obtain a sigma-term $t :: F$ from t_1 such that $[\![t]\!]_{\mathcal{LG}} = \varphi$. Thus, the claim on of-course is always satisfied; thus, we assume it from now on.

In the following, we prove the remaining claims of the lemma by induction on $|F|$. We first consider occurrences of top, tensor and linear implication in Γ, and those of cuts in Π:

- If Γ contains top (respectively, tensor), then we reduce this case to the induction hypothesis (respectively, hypotheses) by the rules $\top$L (respectively, $\otimes$L) and XL.
- Assume an occurrence of cut in Π or linear implication in Γ. By Lemma 3.22, there is the least one among them for $<_\varphi$. Suppose that it is $S \multimap T$ in Γ; write $\Gamma = \Delta, S \multimap T, \Theta$. For each $\boldsymbol{s} \in \varphi$, we write $\boldsymbol{s}@S$ for its justified subsequence that consists of the elements $\boldsymbol{s}(i)$ such that the P-view $\lceil \boldsymbol{s}(1)\boldsymbol{s}(2) \dots \boldsymbol{s}(i) \rceil$ has a move in $[\![S]\!]_{\mathcal{LG}}$, and $\boldsymbol{s}@T$ for the subsequence of $\boldsymbol{s}$ consisting of the remaining elements. Let $\varphi^S := \{ \boldsymbol{s}@S \mid \boldsymbol{s} \in \varphi \}$ and $\varphi^T := \{ \boldsymbol{s}@T \mid \boldsymbol{s} \in \varphi \}$. It suffices to show that φ^S and φ^T do not visit the same element of Γ or Π because then we can reduce this case to the induction hypotheses by the rules $\multimap$L and XL. Because the occurrence $S \multimap T$ is the least one, it suffices to focus on elements of Γ different from linear implication. We can assume that Γ does not have tensor, and that $\lceil \varphi \rceil$ visits an occurrence of of-course in Γ at most once. Thus, the disjointness of φ^S and φ^T holds.
- If an element of Π that is not of-course or with is the least one for $<_\varphi$, then similarly to the above case we reduce this case to the induction hypotheses by the rules Cut and XL.

Let us leave the cases on Φ to the reader since they are just straightforward. $\square$

Definition 3.25 (prime form)**.** A pair (F, φ) of a sigma-sequent F and a 1-cell $\varphi : [\![F]\!]_{\mathcal{LG}}$ in the bicategory $\mathcal{LG}$ is said to be ***prime*** if it satisfies all conditions of the third clause of Lemma 3.24.

We are now ready to prove the main theorem of the present section:

Theorem 3.26 (an intensional surjection)**.** *Given a sigma-sequent F and a 1-cell $\varphi : [\![F]\!]_{\mathcal{LG}}$ in the bicategory $\mathcal{LG}$, there is a sigma-term $t :: F$ such that $[\![t]\!]_{\mathcal{LG}} = \varphi$.*

Proof. Let $F = \Gamma \dashv \Pi \vdash \Phi$. We proceed by induction on $|F|$. By Lemma 3.24, we assume that the pair (F, φ) is prime. In what follows, we proceed by case analysis on Φ. First, assume $\Phi = \boldsymbol{\epsilon}$. If the first computational step of φ happens in an occurrence of cut, then, by $\Phi = \boldsymbol{\epsilon}$ and the definition of 1-cells in $\mathcal{LG}$, the cut does not have an internal additive or an exponential structure, a contradiction. If the first step arises in an occurrence of negation (respectively, with, of-course) in Γ, then we reduce this case to the induction hypothesis (or hypotheses) by the rules $\neg$L (respectively, &L, !D) and XL. The first step is never in an occurrence of one in Γ by the joker axiom. We have considered all cases for $\Phi = \boldsymbol{\epsilon}$. The cases $\Phi = \top$ and $\Phi = 1$ are evident.

Next, suppose $\Phi = S \mathbin{\&} T$. By the definition of 1-cells in $\mathcal{LG}$, every element of Π is with that corresponds to $S \mathbin{\&} T$ via φ. Thus, reduce this case to the induction hypotheses by the rule &R.

Now, assume $\Phi = !S$. If $S = \top$, then $!S = \top$ so that this case coincides with the one $\Phi = \top$. If S is not top, then the linearity of φ together with the definition of 1-cells in $\mathcal{LG}$ implies that Π and Γ contain only of-course, so we can reduce this case to the one $\Phi = S$ by the rule !R.

Finally, assume $\Phi = S \otimes T$. We reduce this case to the induction hypotheses by the rule $\otimes$R similarly to the case where we apply the rules Cut and XL. $\square$

Corollary 3.27 (an intensional biequivalence between logic and combinatorics)**.** *The 2-functor $[\![\_]\!]_{\mathcal{LG}}$ of Proposition 3.16 defines a biequivalence $\mathsf{ILL} \simeq \mathcal{LG}$.*

Proof. The 2-functor is essentially surjective on objects by Corollary 3.7, and fully faithful by Proposition 3.11 and Theorem 3.26. $\square$

Under this biequivalence, the cut-eliminations on sigma-terms can be read as an operation on 1-cells in the bicategory $\mathcal{LG}$, called the *(one-step) hiding operation*, for which we write $\mathcal{H}$.

Notation. Fix a 2-functor $[\![\_]\!]^{\flat}_{\mathcal{LG}} : \mathcal{LG} \to \mathsf{ILL}$ whose object-map is a right inverse of the object-map of the biequivalence $[\![\_]\!]_{\mathcal{LG}}$, and arrow-map is the inverse of the arrow-map of $[\![\_]\!]_{\mathcal{LG}}$.

Definition 3.28 (the hiding operation)**.** The ***(one-step) hiding operation*** is the E-functor $\mathcal{H} : \mathcal{LG} \to \mathcal{LG}$ given by $\mathcal{H}(\mathscr{A}) := \mathscr{A}$ on objects $\mathscr{A}$, and by $\mathcal{H}(\Pi, \varphi) := (\mathcal{H}(\Pi), \mathcal{H}(\varphi)) := (\Pi', \varphi')$ if and only if $[\![\varphi]\!]^{\flat}_{\mathcal{LG}} :: ([\![\mathscr{A} \dashv \Pi \vdash \mathscr{B}]\!]^{\flat}_{\mathcal{LG}}) \to [\![\varphi']\!]^{\flat}_{\mathcal{LG}} :: ([\![\mathscr{A} \dashv \Pi' \vdash \mathscr{B}]\!]^{\flat}_{\mathcal{LG}})$ on 1-cells $(\Pi, \varphi) : \mathscr{A} \to \mathscr{B}$.

The hiding operation $\mathcal{H}$ is much more intricate and cumbersome to describe directly in terms of the combinatorial structures of $\mathcal{LG}$. This is why we have instead defined it in terms of the cut-elimination $\to$ for the sigma-calculus by exploiting the biequivalence $\mathsf{ILL} \simeq \mathcal{LG}$.

Theorem 3.14 is then translated into:

Corollary 3.29 (combinatorial cut-elimination)**.** *Each 1-cell (Π, φ) in the bicategory $\mathcal{LG}$ has a finite sequence $(\Pi, \varphi), \mathcal{H}(\Pi, \varphi), \mathcal{H}^2(\Pi, \varphi), \dots, \mathcal{H}^n(\Pi, \varphi)$ of 1-cells in $\mathcal{LG}$ such that $\mathcal{H}^n(\Pi, \varphi) = (\boldsymbol{\epsilon}, \mathcal{H}^{\omega}(\varphi))$, and $t \to t'$ if and only if $[\![t]\!]_{\mathcal{LG}} \neq [\![t']\!]_{\mathcal{LG}}$ and $\mathcal{H}([\![t]\!]_{\mathcal{LG}}) = [\![t']\!]_{\mathcal{LG}}$ for all sigma-terms t.*

By construction, the hiding operation corresponds precisely to the cut-elimination; also, by Theorem 3.26, it is defined on *all* 1-cells in $\mathcal{LG}$. Consequently, it *completely* captures the cut-elimination: $t \to t'$ if and only if $[\![t]\!]_{\mathcal{LG}} \neq [\![t']\!]_{\mathcal{LG}}$ and $\mathcal{H}([\![t]\!]_{\mathcal{LG}}) = [\![t']\!]_{\mathcal{LG}}$ for all sigma-terms t. To the best of our knowledge, this is the first syntax-free, non-inductive recast of cut-elimination.

Because the bicategories CLL and $\mathcal{LG}_{\neg\neg}$ (respectively, IL and $\mathcal{LG}_{!}$, CL and $\mathcal{LG}_{!?}$) are obtained from ILL and $\mathcal{LG}$, respectievly, through the same (co-)Kleisli construction, Corollary 3.29 extends to this pair of logic and combinatorics too. As a result, we have reduced the proof theory of these propositional logics to the study of combinatorics, including their dynamics and intensionality.

Another immediate consequence of Corollary 3.27 of theoretical interest is:

Corollary 3.30 (linear winning implies flat innocence)**.** *Every linearly winning strategy is flatly innocent.*

4 Combinatorial higher-order computation

This last section shows that the bicategory $\mathcal{G}_!$ admits a model computation that can simulate the higher-order functional programming language *PCF* [Sco93, Plo77]. This *PCF-completeness* of $\mathcal{G}_!$ verifies that our combinatorics induces a quite general class of higher-order computation.

The main theorem of Yamada [Yam19] states that strategies *presentable by finitely presentable strategies* can simulate PCF. Our PCF-completeness significantly improves this theorem by proving that finitely presentable ones suffice to simulate PCF. This advance is due to our combinatorial recast of games: The recast dispenses with the computation on infinitary tags in [Yam19].

The improvement is not only on the theorem itself but also on the underlying framework: The preceding approach [Yam19] entails extremely intricate constructions on finitely presentable strategies that present strategies interpreting PCF; in contrast, our method is not bothered by such constructions at all. This simplification is important for our framework to be *usable*.

Remark. Another, more conceptual advance is that, by dispensing with the finitely presentable strategies that *extrinsically* and *non-canonically* encode strategies interpreting PCF in [Yam19], and more generally by freeing from any syntactic encodings typical in existing models of computation, all structures for computation in $\mathcal{G}_!$ are *intrinsic* and thus *canonical* to $\mathcal{G}_!$. Note that $\mathcal{G}_!$ consists of semantic, in particular syntax-free and non-inductive, objects. Hence, if the standard formalism in mathematics, or the category of sets, is replaced with $\mathcal{G}_!$, then the present result resolves the extrinsic, non-canonical nature of computability raised in [Abr14, §1.2].

To establish the PCF-completeness of 1-cells in $\mathcal{G}_!$, let us begin with showing that strategies modelling *atomic* terms in PCF are all finitely presentable:

Notation. In the following examples, we use the superscripts $(\_)'$, $(\_)''$, etc., as informal tags on combinatorial arenas and their vertices.

Example 4.1. Recall the finitely presentable strategy $\mathrm{succ} : (\mathscr{N} \dashv\vdash \mathscr{N}')$ given in Example 2.51. This computation also forms a finitely presentable strategy on the combinatorial sequent $!\mathscr{N} \dashv\vdash \mathscr{N}'$. Thus, the pair $(\boldsymbol{\epsilon}, \mathrm{succ})$ is a normalised 1-cell $\mathscr{N} \to \mathscr{N}$ in the bicategory $\mathcal{G}_!$. It computes the successor $\mathbb{N} \to \mathbb{N}$ in the sense that $\mathrm{succ}(\underline{n}^\dagger) \simeq \underline{n+1}$ holds for all $n \in \mathbb{N}$.

Example 4.2. It is not hard to see that there are no normalised 1-cells $\mathscr{N} \to \mathscr{N}$ in $\mathcal{G}_!$ that play as the predecessor $\mathbb{N} \to \mathbb{N}$. This difference indicates that predecessor is slightly more complex than successor according to our combinatorial framework.

To construct a 1-cell $\mathscr{N} \to \mathscr{N}$ for the predecessor, we define strategies $\mathrm{pred}_+ : (!\mathscr{N} \dashv\vdash \mathscr{N}'_{\neg\neg})$ and $\mathrm{pred}\_ : (!\mathscr{N}_{\neg\neg} \dashv\vdash \mathscr{N}')$, where $\mathscr{N}_{\neg\neg} := !(\bot_{[q]} \multimap_{[p]} \bot_{[y]}) \otimes \bot_{[n]} \multimap_{[o]} \neg_{[i]}\neg_{[j]}\bot_{[\hat{q}]}$, by

$$\begin{aligned}\mathrm{pred}_+ := \mathrm{Pref}(\{\, &\dashv\vdash o'i'.\vec{q}_0.\vec{\mathrm{no}}.\vec{\mathrm{no}}'\,\}\\ &\cup \{\, \dashv\vdash o'i'.\vec{q}_0.\vec{\mathrm{yes}}_0.j'.(\vec{q}'_0.\vec{q}_1.\vec{\mathrm{yes}}_1.\vec{\mathrm{yes}}'_0)(\vec{q}'_1.\vec{q}_2.\vec{\mathrm{yes}}_2.\vec{\mathrm{yes}}'_1)\dots(\vec{q}'_n.\vec{q}_{n+1}.\vec{\mathrm{yes}}_{n+1}.\vec{\mathrm{yes}}'_n).\vec{q}'_{n+1}.\vec{q}_{n+2}.\vec{\mathrm{no}}.\vec{\mathrm{no}}' \mid n \in \mathbb{N}\,\})^{\mathrm{Even}}\\ \mathrm{pred}\_ := \mathrm{Pref}(\{\, &\dashv\vdash \vec{q}'_0.i.j.\vec{q}_0.(\vec{\mathrm{yes}}_0.\vec{\mathrm{yes}}'_0.\vec{q}'_1.\vec{q}_1)(\vec{\mathrm{yes}}_1.\vec{\mathrm{yes}}'_1.\vec{q}'_2.\vec{q}_2)\dots(\vec{\mathrm{yes}}_n.\vec{\mathrm{yes}}'_n.\vec{q}'_{n+1}.\vec{q}_{n+1}).\vec{\mathrm{no}}.\vec{\mathrm{no}}' \mid n \in \mathbb{N}\,\})^{\mathrm{Even}}.\end{aligned}$$

We then define $\mathrm{pred} := \mathrm{pred}_+ \parr \mathrm{pred}\_$ on $!\mathscr{N} \dashv !\mathscr{N}_{\neg\neg} \vdash \mathscr{N}$. It is easy to see that pred_+ and $\mathrm{pred}\_$ are finitely presentable, and thus so is pred. Hence, the pair $(\mathscr{N}_{\neg\neg}, \mathrm{pred})$ is a 1-cell in $\mathcal{G}_!$. It indeed plays as the predecessor: $\mathrm{pred}(\underline{n+1}^\dagger) \simeq \underline{n}$ for all $n \in \mathbb{N}$ and $\mathrm{pred}(\underline{0}^\dagger) \simeq \underline{0}$ hold.

Example 4.3. We define the combinatorial arena $\mathscr{B}$, called the ***boolean combinatorial arena***, by $\mathscr{B} := \bot_{[\mathrm{tt}]} \otimes \bot_{[\mathrm{ff}]} \multimap_{[o]} \bot_{[q]}$, and the strategies $\underline{\mathrm{tt}}, \underline{\mathrm{ff}} : \mathscr{B}$ by $\underline{\mathrm{tt}} := \{\boldsymbol{\epsilon}, oq.\mathrm{tt}\}$ and $\underline{\mathrm{ff}} := \{\boldsymbol{\epsilon}, oq.\mathrm{ff}\}$.

The ***(binary) conditional*** on a given combinatorial arena $\mathscr{A}$ refers to the strategy $\mathrm{case}_{\mathscr{A}} : !\mathscr{A}, !\mathscr{A}', !\mathscr{B}'' \dashv\vdash \mathscr{A}'''$ defined by

$$\mathrm{case}(\mathscr{A})_! := \mathrm{Pref}(\{\, \vec{a}'''_1.o''q''.\mathrm{tt}''.\vec{a}_1.\boldsymbol{s} \mid \vec{a}'''_1.\vec{a}_1.\boldsymbol{s} \in \mathrm{der}_{\mathscr{A}}\,\} \cup \{\, \vec{a}'''_1.o''q''.\mathrm{ff}''.\vec{a}'_1.\boldsymbol{t} \mid \vec{a}'''_1.\vec{a}'_1.\boldsymbol{t} \in \mathrm{der}_{\mathscr{A}}\,\})^{\mathrm{Even}}.$$

It is easy to see that this strategy is finitely presentable, and it plays as the binary conditional: $\mathrm{case}_{\mathscr{A}}(\varphi_1^\dagger \otimes \varphi_2^\dagger \otimes \underline{\mathrm{tt}}^\dagger) \simeq \varphi_1$ and $\mathrm{case}_{\mathscr{A}}(\varphi_1^\dagger \otimes \varphi_2^\dagger \otimes \underline{\mathrm{ff}}^\dagger) \simeq \varphi_2$ for all strategies $\varphi_1, \varphi_2 : (\Gamma \dashv \Pi \vdash \mathscr{A})$.

In addition, the ***ifzero*** refers to the strategy $\mathrm{ifzero} : !\mathscr{N} \dashv\vdash \mathscr{B}$ defined by

$$\mathrm{ifzero} := \mathrm{Pref}(\{\, \dashv\vdash oq.\vec{q}_0.\vec{\mathrm{no}}.\mathrm{tt}\,\} \cup \{\, \dashv\vdash oq.\vec{q}_0.\vec{\mathrm{yes}}_0.\mathrm{ff}\,\})^{\mathrm{Even}}.$$

It is clear that this strategy is finitely presentable, and its computation checks whether a given winning strategy $\underline{n} : \mathscr{N}$ is $\underline{0}$ by $\mathrm{ifzero}(\underline{n}) \simeq \underline{\mathrm{tt}}$ if $n = 0$, and $\mathrm{ifzero}(\underline{n}) \simeq \underline{\mathrm{ff}}$ otherwise.

Example 4.4. The ***fixed-point combinator*** on a combinatorial arena $\mathscr{A}$ is the strategy $\mathrm{fix}_{\mathscr{A}} : !(\mathscr{A} \Rightarrow \mathscr{A}') \dashv\!\vdash \mathscr{A}''$ that computes, loosely speaking, by playing as the dereliction between $\mathscr{A}'$ and $\mathscr{A}''$ as well as between $\mathscr{A}$ and $\mathscr{A}'$; see [Hyl97, §2.3.3] or [Yam19, Example 75] for the details.

This strategy calculates the *fixed-point* of a given strategy $\varphi : \mathscr{A} \Rightarrow \mathscr{A}$ in the sense that $\varphi(\mathrm{fix}_{\mathscr{A}}(\varphi^{\dagger})^{\dagger}) \simeq \varphi$ holds. Because the fixed-point combinator plays essentially as the two derelictions, it is finitely presentable. The significance of this finite presentability is that it completely dispenses with the intricate computation on tags for exponentials in [Yam19, Example 75].

One way to enumerate strategies definable by PCF, up to the hiding equivalence, is to start from $\mathrm{der}_{\mathscr{A}}$, $\underline{0}_{\mathscr{A}}$, succ, pred, $\mathrm{case}_{\mathscr{A}}$, ifzero and $\mathrm{fix}_{\mathscr{A}}$, called *PCF-atomic* strategies, where $\mathscr{A}$ is generated from $\mathscr{N}$, $\mathscr{B}$ and/or $\top$ by product and/or implication, and apply parallel composition, transpose, pairing and promotion; see the proof of [Yam19, Theorem 81] (n.b., a variant of parallel composition is called *concatenation* and written ‡ in [Yam19]). Thus, it suffices to show that PCF-atomic ones are finitely presentable, and the constructions preserve finite presentability.

We have accomplished the first task by the above examples; the second task has been completed by Propositions 2.47 and 2.50. Thus, the main theorem of the present section follows:

Theorem 4.5 (a combinatorial model of higher-order computation)**.** *Each strategy $\varphi : (\Gamma \dashv\!\vdash \mathscr{A})$ definable by PCF has a 1-cell (Π, κ) in the bicategory $\mathcal{G}_!$ that satisfies $\mathcal{H}^{\omega}(\kappa) = \varphi$.*

Most part of the long article [Yam19] is dedicated to its main theorem [Yam19, Theorem 81], which states that every strategy definable by PCF is presentable by a finitely presentable strategy. Our theorem improves this result by showing that in our combinatorial setting finitely presentable strategies suffice to simulate PCF. Moreover, by dispensing with the strategies presenting PCF-definable ones, our proof is much simpler and more straightforward than the previous one.

Although there is no doubt about the computability of finitely presentable strategies, one may wonder whether it is *maximal.* In particular, because a standard idea in the literature [HO00] is to define an innocent strategy to be computable or ***recursive*** if its P-view map is calculable by a Turing machine, do we gain a stronger notion of computability by replacing finite presentability of innocent strategies with recursiveness? The answer is negative:

Corollary 4.6 (completeness of finite presentability)**.** *Every recursive innocent strategy φ on a combinatorial sequent $!\mathscr{A} \dashv\!\vdash \mathscr{B}$ has a 1-cell (Ξ, κ) in the bicategory $\mathcal{G}_!$ such that $\kappa \simeq \varphi$.*

Proof (sketch). Fix an effective encoding $\# : \lceil \mathcal{P}_{!\mathscr{A} \dashv\!\vdash \mathscr{B}} \rceil \rightarrowtail \mathbb{N}$ with its range decidable, and its inverse effective. Because PCF-completeness implies Turing completeness, Theorem 4.5 finds a finitely presentable strategy κ_{φ} on some combinatorial sequent $!\mathscr{N} \dashv \Pi \vdash \mathscr{N}$ that computes the P-view map f_{φ} of φ in the sense that $\kappa_{\varphi}(\overline{\#\lceil \boldsymbol{s}\vec{o} \rceil}) = \overline{\#\lceil \boldsymbol{s}\vec{o}\vec{p} \rceil}$ if and only if $f_{\varphi}(\lceil \boldsymbol{s}\vec{o} \rceil) = \lceil \boldsymbol{s}\vec{o}\vec{p} \rceil$ for all odd-length P-views $\lceil \boldsymbol{s}\vec{o} \rceil$ and P-moves $\vec{p}$ in $!\mathscr{A} \dashv\!\vdash \mathscr{B}$. Let $\tilde{\kappa}_{\varphi} := \lambda(\kappa_{\varphi}) : (\dashv \Pi \vdash !\mathscr{N} \multimap \mathscr{N})$.

For the same reason, there is a finitely presentable one μ on some $!\mathscr{A}, !(\mathscr{N} \Rightarrow \mathscr{N}) \dashv !\mathscr{N}, \Sigma \vdash \mathscr{B}$ that computes as follows. At each odd-length position such that the last O-move $\vec{o}$ is in $!\mathscr{A}$ or $\mathscr{B}$, μ first records $\vec{o}$ and its justifier $\vec{\jmath}$ on a new copy of $\mathscr{N}$ in $!\mathscr{N}$; in this way, κ_{φ} sees each P-view in $!\mathscr{A} \dashv\!\vdash \mathscr{B}$ even though it is flatly innocent. Next, μ reads off the current P-view $\lceil \boldsymbol{s}\vec{o} \rceil$ in $!\mathscr{A} \dashv\!\vdash \mathscr{B}$ from what has been played in $!\mathscr{N}$, and computes in $!(\mathscr{N} \Rightarrow \mathscr{N})$ by playing as $\overline{\#\lceil \boldsymbol{s}\vec{o} \rceil}$ in the domain of $\mathscr{N} \Rightarrow \mathscr{N}$. Finally, μ reads off the output given in the codomain of $\overline{\mathscr{N} \Rightarrow \mathscr{N}}$ and makes the corresponding P-move in $!\mathscr{A}$ or $\mathscr{B}$, if any; it does not make the next P-move if there is no corresponding P-move. Then, define $\Xi := \Pi, !(\mathscr{N} \Rightarrow \mathscr{N}), !\mathscr{N}, \Sigma$ and $\kappa := \tilde{\kappa}_{\varphi}^{\dagger} \between \mu$. □

Note that an argument similar to that of the proof implies:

Corollary 4.7 (completeness of flat innocence)**.** *If φ is an innocent strategy on a combinatorial sequent $\Gamma \dashv \Pi \vdash \Phi$, then there is a flatly innocent strategy ℓ on some combinatorial sequent $\Gamma \dashv \Pi, \Sigma \vdash \Phi$ that coincides with φ up to the hiding equivalence, i.e., $\varphi \simeq \ell$.*

By Church-Turing thesis, the corollaries imply that finite presentability defines the maximal or *canonical* notion of computability of innocent strategies in our setting. A main advantage of finite presentability over recursiveness is its fine analysis of computational complexity, which is explicitly exhibited in the intensionality of a combinatorial sequent. Also, the former is intrinsic to the structure of strategies and free from the extrinsic manipulation of symbols on tapes.

Acknowledgments

The author acknowledges the financial support from Funai Overseas Scholarship, and also he is grateful to Samson Abramsky for fruitful discussions.

References

[A$^+$97] Samson Abramsky et al., *Semantics of interaction: An introduction to game semantics*, Semantics and Logics of Computation, Publications of the Newton Institute (1997), 1–31.

[Abr93] Samson Abramsky, *Computational interpretations of linear logic*, Theoretical computer science **111** (1993), no. 1-2, 3–57.

[Abr14] ———, *Intensionality, definability and computation*, Johan van Benthem on Logic and Information Dynamics, Springer, 2014, pp. 121–142.

[ABS06] Marcelo Aguiar, Nantel Bergeron, and Frank Sottile, *Combinatorial hopf algebras and generalized dehn–sommerville relations*, Compositio Mathematica **142** (2006), no. 1, 1–30.

[Acz86] Peter Aczel, *The type theoretic interpretation of constructive set theory: inductive definitions*, Studies in Logic and the Foundations of Mathematics, vol. 114, Elsevier, 1986, pp. 17–49.

[AHM98] Samson Abramsky, Kohei Honda, and Guy McCusker, *A fully abstract game semantics for general references*, Logic in Computer Science, 1998. Proceedings. Thirteenth Annual IEEE Symposium on, IEEE, 1998, pp. 334–344.

[AJ94] Samson Abramsky and Radha Jagadeesan, *Games and full completeness for multiplicative linear logic*, The Journal of Symbolic Logic **59** (1994), no. 02, 543–574.

[AJM00] Samson Abramsky, Radha Jagadeesan, and Pasquale Malacaria, *Full abstraction for PCF*, Information and Computation **163** (2000), no. 2, 409–470.

[AJV15] Samson Abramsky, Radha Jagadeesan, and Matthijs Vákár, *Games for dependent types*, Automata, Languages, and Programming, Springer, Berlin, Heidelberg, 2015, pp. 31–43.

[AM97] Samson Abramsky and Guy McCusker, *Linearity, sharing and state: a fully abstract game semantics for Idealized Algol with active expressions*, Algol-like languages, Springer, 1997, pp. 297–329.

[AM99a] ———, *Full abstraction for Idealized Algol with passive expressions*, Theoretical Computer Science **227** (1999), no. 1, 3–42.

[AM99b] ______, *Game semantics*, Computational logic, Springer, 1999, pp. 1–55.

[AM99c] Samson Abramsky and P-A Mellies, *Concurrent games and full completeness*, Proceedings. 14th Symposium on Logic in Computer Science (Cat. No. PR00158), IEEE, 1999, pp. 431–442.

[B$^+$84] Hendrik Pieter Barendregt et al., *The lambda calculus*, vol. 3, North-Holland, Amsterdam, 1984.

[Bac59] John W Backus, *The syntax and semantics of the proposed international algebraic language of the zurich acm-gamm conference*, Proceedings of the International Conference on Information Processing, 1959 (1959).

[Bar06] Michael Barr, **-autonomous categories*, vol. 752, Springer, 2006.

[BBDPH93] Nick Benton, Gavin Bierman, Valeria De Paiva, and Martin Hyland, *A term calculus for intuitionistic linear logic*, International Conference on Typed Lambda Calculi and Applications, Springer, 1993, pp. 75–90.

[Bén67] Jean Bénabou, *Introduction to bicategories*, Reports of the midwest category seminar, Springer, 1967, pp. 1–77.

[Bie95] Gavin M Bierman, *What is a categorical model of intuitionistic linear logic?*, International Conference on Typed Lambda Calculi and Applications, Springer, 1995, pp. 78–93.

[Bie99] ______, *A classical linear λ-calculus*, Theoretical Computer Science **227** (1999), no. 1-2, 43–78.

[BP96] Andrew Barber and Gordon Plotkin, *Dual intuitionistic linear logic*, University of Edinburgh, Department of Computer Science, Laboratory for . . . , 1996.

[CH10] Pierre Clairambault and Russ Harmer, *Totality in arena games*, Annals of Pure and Applied Logic **161** (2010), no. 5, 673–689.

[Cla21] Pierre Clairambault, *A tale of additives and concurrency in game semantics.*

[Coq95] Thierry Coquand, *A semantics of evidence for classical arithmetic*, The Journal of Symbolic Logic **60** (1995), no. 1, 325–337.

[CR79] Stephen A Cook and Robert A Reckhow, *The relative efficiency of propositional proof systems*, The journal of symbolic logic **44** (1979), no. 1, 36–50.

[DJS95] Vincent Danos, Jean-Baptiste Joinet, and Harold Schellinx, *Lkq and lkt: sequent calculi for second order logic based upon dual linear decompositions of classical implication*, Advances in Linear Logic **222** (1995), 211–224.

[DJS97] ______, *A new deconstructive logic: Linear logic*, The Journal of Symbolic Logic **62** (1997), no. 3, 755–807.

[DM68] Augustus De Morgan, *Review of jm wilson's 'elementary geometry'*, The Athenaeum **2** (1868), no. 2125, 71–73.

[Dum93] Michael Dummett, *The logical basis of metaphysics*, Harvard university press, 1993.

[Fre79] Gottlob Frege, *Begriffsschrift, a formula language, modeled upon that of arithmetic, for pure thought*, From Frege to Gödel: A source book in mathematical logic **1931** (1879), 1–82.

[Gen35] Gerhard Gentzen, *Untersuchungen über das logische schließen. i*, Mathematische zeitschrift **39** (1935), no. 1, 176–210.

[Gen36] ———, *Die widerspruchsfreiheit der reinen zahlentheorie*, Mathematische annalen **112** (1936), no. 1, 493–565.

[Gir87] Jean-Yves Girard, *Linear logic*, Theoretical computer science **50** (1987), no. 1, 1–101.

[Gir89] ———, *Geometry of interaction 1: Interpretation of system f*, Studies in Logic and the Foundations of Mathematics, vol. 127, Elsevier, 1989, pp. 221–260.

[GL87] Jean-Yves Girard and Yves Lafont, *Linear logic and lazy computation*, International Joint Conference on Theory and Practice of Software Development, Springer, 1987, pp. 52–66.

[Göd29] Kurt Gödel, *Uber die vollständigkeit des logikkalküls doctoral thesis, d1. 736 33pp, university of vienna (1929). reprinted in feferman, s, ed. gödel collected works, volume 1, publications 1929-1936*, 1929.

[Gun92] Carl A Gunter, *Semantics of programming languages: Structures and techniques*, MIT press, Cambridge, MA, 1992.

[Hey30] Arend Heyting, *Die formalen regeln der intuitionistischen logik*, Sitzungsbericht PreuBische Akademie der Wissenschaften Berlin, physikalisch-mathematische Klasse II (1930), 42–56.

[Hil26] David Hilbert, *Uber das unendliche*, Mathematische Annalen **95** (1926), 161–190.

[Hil31] ———, *Die grundlegung der elementaren zahlenlehre*, Mathematische Annalen **104** (1931), no. 1, 485–494.

[HO93] J Martin E Hyland and C-H Luke Ong, *Fair games and full completeness for multiplicative linear logic without the mix-rule*, preprint **190** (1993).

[HO00] J Martin E Hyland and C-HL Ong, *On Full Abstraction for PCF: I, II, and III*, Information and computation **163** (2000), no. 2, 285–408.

[Hop64] Heinz Hopf, *Über die topologie der gruppen-mannigfaltigkeiten und ihrer verallgemeinerungen: Annals of mathematics vol. 42, no. 1, 1941 received by compositio math., august 23, 1939*, Selecta Heinz Hopf: Herausgegeben zu seinem 70. Geburtstag von der Eidgenössischen Technischen Hochschule Zürich, Springer, 1964, pp. 119–151.

[Hug06] Dominic JD Hughes, *Proofs without syntax*, Annals of Mathematics (2006), 1065–1076.

[Hyl97] Martin Hyland, *Game semantics*, Semantics and logics of computation, vol. 14, Cambridge University Press, New York, 1997, p. 131.

[JR79] SA Joni and G-C Rota, *Coalgebras and bialgebras in combinatorics*, Studies in Applied Mathematics **61** (1979), no. 2, 93–139.

[Lai97] James Laird, *Full abstraction for functional languages with control*, Logic in Computer Science, 1997. LICS'97. Proceedings., 12th Annual IEEE Symposium on, IEEE, 1997, pp. 58–67.

[Lau04] Olivier Laurent, *Polarized games*, Annals of Pure and Applied Logic **130** (2004), no. 1-3, 79–123.

[Lau05] ———, *Syntax vs. semantics: a polarized approach*, Theoretical Computer Science **343** (2005), no. 1-2, 177–206.

[LS88] Joachim Lambek and Philip J Scott, *Introduction to Higher-order Categorical Logic*, vol. 7, Cambridge University Press, 1988.

[McC98] Guy McCusker, *Games and full abstraction for a functional metalanguage with recursive types*, Springer Science & Business Media, London, 1998.

[Mel05] P-A Mellies, *Asynchronous games 4: A fully complete model of propositional linear logic*, 20th Annual IEEE Symposium on Logic in Computer Science (LICS'05), IEEE, 2005, pp. 386–395.

[Mel09] Paul-André Mellies, *Categorical semantics of linear logic*, Panoramas et syntheses **27** (2009), 15–215.

[MO03] Andrzej S. Murawski and C-HL Ong, *Exhausting strategies, joker games and full completeness for imll with unit*, Theoretical Computer Science **294** (2003), no. 1-2, 269–305.

[MT10] Paul-André Melliès and Nicolas Tabareau, *Resource modalities in tensor logic*, Annals of Pure and Applied Logic **161** (2010), no. 5, 632–653.

[Nic94] Hanno Nickau, *Hereditarily sequential functionals*, International Symposium on Logical Foundations of Computer Science, Springer, 1994, pp. 253–264.

[Plo77] Gordon D. Plotkin, *Lcf considered as a programming language*, Theoretical computer science **5** (1977), no. 3, 223–255.

[Pra71] Dag Prawitz, *Ideas and results in proof theory*, Studies in Logic and the Foundations of Mathematics, vol. 63, Elsevier, 1971, pp. 235–307.

[Sch93] William R Schmitt, *Hopf algebras of combinatorial structures*, Canadian Journal of Mathematics **45** (1993), no. 2, 412–428.

[Sco70] Dana Scott, *Outline of a mathematical theory of computation*, Oxford University Computing Laboratory, Programming Research Group Oxford, 1970.

[Sco93] Dana S Scott, *A type-theoretical alternative to iswim, cuch, owhy*, Theoretical Computer Science **121** (1993), no. 1-2, 411–440.

[See87] Robert AG Seely, *Linear logic,*-autonomous categories and cofree coalgebras*, Ste. Anne de Bellevue, Quebec: CEGEP John Abbott College, 1987.

[Sel01] Peter Selinger, *Control categories and duality: on the categorical semantics of the lambda-mu calculus*, Mathematical Structures in Computer Science **11** (2001), no. 2, 207–260.

[Tro91] Anne Sjerp Troelstra, *Lectures on linear logic.*

[TS00] Anne Sjerp Troelstra and Helmut Schwichtenberg, *Basic proof theory*, no. 43, Cambridge University Press, 2000.

[Tur37] Alan M Turing, *On computable numbers, with an application to the entscheidungsproblem*, Proceedings of the London mathematical society **2** (1937), no. 1, 230–265.

[YA20] Norihiro Yamada and Samson Abramsky, *Dynamic game semantics*, Mathematical Structures in Computer Science **30** (2020), no. 8, 892–951.

[Yam19] Norihiro Yamada, *Game semantics of martin-löf type theory*, arXiv preprint arXiv:1905.00993 (2019), accepted for publication by *Mathematical Structures in Computer Science.*