Game semantics of universes

Norihiro Yamada

yamad041@umn.edu
School of Mathematics
University of Minnesota

March 25, 2022

Abstract

This work extends the present author’s computational game semantics of Martin-Löf type theory to the cumulative hierarchy of universes. This extension completes game semantics of all standard types of Martin-Löf type theory for the first time in the 30 years history of modern game semantics. As a result, the powerful combinatorial reasoning of game semantics becomes available for the study of universes and types generated by them. A main challenge in achieving game semantics of universes comes from a conflict between identity types and universes: Naive game semantics of the encoding of an identity type by a universe induces a decision procedure on the equality between functions, a contradiction to a well-known fact in recursion theory. We overcome this problem by novel games for universes that encode games for identity types without deciding the equality.

Contents

1 Introduction
1.1 Martin-Löf type theory and the meaning explanation 2
1.2 Game semantics of Martin-Löf type theory 2
1.3 Examples of games and strategies 3
1.4 Martin-Löf’s universes 6
1.5 The problem: how to encode games for identity types by strategies 6
1.6 Our solution: encoding without deciding 8
1.7 Lifting to the cumulative hierarchy of universes 9
1.8 Main results 9
1.9 Our contributions and related work 11
1.10 The structure of the present article 11

2 Review: game semantics of Martin-Löf type theory
2.1 Games and strategies 12
2.2 Game semantics of Martin-Löf type theory 15

3 Game semantics of universes
3.1 Universe predicate games 20
3.2 Computational game semantics of the cumulative hierarchy of universes 22

4 Corollaries
4.1 Effectivity of game semantics 27
4.2 Independence of equality reflection 27
4.3 Independence of Markov’s principle 28

5 Conclusion and future work 28

More extensional, domain and realisability semantics of universes has been established though [Pa93, Str12, BL18].
1 Introduction

For this introduction, we assume that the reader is familiar with the syntax of dependent type theories and universes \cite{Ho97}, but not with game semantics \cite{A+97, Hyl97}.

1.1 Martin-Löf type theory and the meaning explanation

On the one hand, formal systems \cite{Sho67} are a class of symbolic or syntactic formalisations of mathematics, and constructive mathematics \cite{TvD88} is a family of computational or constructive schools in mathematics. On the other hand, semantics of a formal system is an assignment of syntax-free objects to syntactic objects of the formal system, where the former serves as the ‘meaning’ or interpretation of the latter.

Martin-Löf type theory (MLTT) \cite{ML75, ML84, ML98} is a prominent formal system for constructive mathematics, and it is comparable to axiomatic set theory \cite{Zer08, Fra22} for classical mathematics. The fundamental idea of MLTT is to regard (mathematical) objects and proofs in constructive mathematics uniformly as computations in an informal sense, and MLTT is a syntactic formalisation of this beautiful idea \cite{ML82}. Hence, objects and proofs in MLTT are unified into terms, where formulas are called types. This standard yet informal semantics of MLTT is called the meaning explanation \cite{DP16, §5}.

Nevertheless, by its informal nature, the meaning explanation cannot serve as a mathematically firm ground to analyse, justify or develop MLTT. Besides, MLTT is an intricate formal system that inevitably contains superficial syntactic details, which makes it difficult to study the meta-theory of MLTT.

1.2 Game semantics of Martin-Löf type theory

This problem calls for mathematical semantics \cite{Gun92} of MLTT that faithfully formalises the meaning explanation, abstracting the inessential syntactic details, and advances the meta-theoretical study of MLTT. Motivated in this way, the present author has established game semantics of MLTT \cite{Yam22}.

Game semantics \cite{A+97, Hyl97} is a class of mathematical semantics that interprets types by games between Player (or a mathematician) and Opponent (or an oracle), and terms by strategies for Player on how to play on games. Games are a class of rooted directed forests, and strategies are algorithms for Player on how to walk on (or play) games alternately with Opponent in such a way that it is Player’s win.

We write a walk or play in a game by a potentially infinite sequence of finite sequences

\[ \epsilon, m_1, m_1m_2, m_1m_2m_3, \ldots, \]

where \( \epsilon \) is the empty sequence, each element or move \( m_i \) is a vertex of the game, and each sequence or position \( m_1m_2\ldots m_n \) is a finite path from the root in the game. By convention, the first move \( m_1 \) is always made by Opponent, and then Player and Opponent alternately make moves. Thus, the moves \( m_{2i+1} \) are made by Opponent, and the other ones \( m_{2i} \) by Player \((i \in \mathbb{N})\). Because a strategy describes the next move to be made by Player, if any, we describe its computational step by the partial function

\[ m_1 \mapsto m_2, m_1m_2m_3 \mapsto m_4, \ldots. \]

The game semantics of MLTT formalises the meaning explanation syntax-independently and intuitively by interpreting terms as strategies or interactive computations between Player and Opponent. In addition, the game semantics turns out to be a highly effective tool for the meta-theoretic study of MLTT; e.g., it verifies the independence of Markov’s principle \cite{Yam22}, which is not possible by most other mathematical semantics of MLTT such as Hyland’s effective topos \cite{Hyl82}. The point is that game semantics is unique in its interpretation of terms by strategies or intensional processes, while other mathematical semantics interprets terms by extensional objects such as functions. Because terms are also intensional objects, computing in a step-by-step fashion, game semantics achieves a very tight correspondence between terms and strategies, which makes itself an exceptionally powerful tool for the study of formal systems.
1.3 Examples of games and strategies

In the following, let us see some simple examples of games and strategies as a preparation for §1.5. For instance, the game \( N \) of natural numbers is the rooted tree (which is infinite in width)

![Rooted tree](image)

in which a play starts with Opponent’s move or question \( q \) (‘What is your number?’) and ends with Player’s move or answer \( n \in \mathbb{N} \) (‘My number is \( n \)). This natural number game \( N \) is not very different from the set \( \mathbb{N} \) of all natural numbers, and there is a much more intensional game for natural numbers \([Yam19]\). However, the game \( N \) is simpler and suffices for our purpose. A strategy \( \sigma \) on \( N \), written \( \sigma : N \), corresponding to the number \( \sigma \in \mathbb{N} \) for instance, is the map \( q \mapsto \sigma(q) \).

In the rest of this introduction, we describe games by listing their positions. For example, the set of all positions of \( N \) is \( \{\epsilon, q\} \cup \{qn | n \in \mathbb{N}\} \).

There is a binary construction \& on games, called product, which forms binary product in the category of games and strategies. The product \( A \& B \) of games \( A \) and \( B \) are simply the disjoint union of \( A \) and \( B \). In other words, a position of \( A \& B \) is either a position of \( A \) or \( B \). For instance, a maximal position of the product \( N \& N \) of the game \( N \) and itself is either of the following forms:

\[
N_{[0]}^{[0]} \& N_{[1]}^{[0]} \quad N_{[0]}^{[1]} \& N_{[1]}^{[1]}
\]

where \( n, m \in \mathbb{N} \), and the subscripts \([i]\) \((i = 0, 1)\) are arbitrary tags to distinguish the two copies of \( N \). We often omit the tags \([i]\) when it does not bring confusion. We write \((n, m)\) for the strategy on \( N \& N \) that plays as in the above diagrams, which forms the pairing of the strategies \( n, m : N \).

Another central construction \( \rightarrow \), called linear implication, captures the notion of linear functions, i.e., functions that consume exactly one input to produce an output. A position of the linear implication \( A \rightarrow B \) between \( A \) and \( B \) is an interleaving mixture of a position of \( A \) and a position of \( B \) such that

1. The first element of the position must be a move of \( B \);
2. A change of the \( AB \)-parity in the position must be made by Player.

For example, a typical position of the linear implication \( N \rightarrow N \) is

\[
N_{[0]} \rightarrow N_{[1]}
\]

where \( n, m \in \mathbb{N} \), which can be read as follows:

1. Opponent’s question \( q_{[1]} \) for an output (‘What is your output?’);
2. Player’s question \( q_{[0]} \) for an input (‘Wait, what is your input?’);
3. Opponent’s answer, say, \( n_{[0]} \), to \( q_{[0]} \) (‘OK, here is an input \( n \));
4. Player’s answer, say, \( m_{[1]} \), to \( q_{[1]} \) (‘Alright, the output is then \( m \)).

This play corresponds to any linear function \( \mathbb{N} \rightarrow \mathbb{N} \) that maps \( n \mapsto m \). The strategy succ on \( N \rightarrow N \) for the successor function is the map \( q_{[1]} \mapsto q_{[0]}, q_{[1]}n_{[0]} \mapsto n + 1_{[1]} \), or diagrammatically

\[
N_{[0]}^{\text{succ}} \rightarrow N_{[1]}
\]

\[
q_{[0]} \quad n_{[0]} \quad n + 1_{[1]}
\]

\[\]
Let us remark here that the following play, which corresponds to a constant linear function that maps $x \mapsto m$ for all $x \in \mathbb{N}$, is also possible: $q[1] \mapsto m[1]$. Thus, strictly speaking, $A \rightarrow B$ is the game of affine functions from $A$ to $B$, but we follow the standard convention to call $\rightarrow$ linear implication.

However, the linear implication $N \& N \rightarrow N$ cannot accommodate strategies that compute binary functions such as addition because maximal positions of this game are either of the following forms:

$$
\begin{array}{c}
N \& N \rightarrow N \\
q \quad q \\
n \quad m
\end{array}
$$

The unary construction $!$ on games, called exponential, addresses this problem by defining the desired game $A \Rightarrow B$ for ordinary (not necessarily linear) functions from $A$ to $B$ by $A \Rightarrow B := !A \rightarrow B$. This idea comes from linear logic [Gir87]. A position of the exponential $!A$ is an interleaving mixture of a finite number of positions of $A$ such that a switch between different copies of positions of $A$ inside $!A$ must be made by Opponent. For instance, the exponential $!(N \& N)$ accommodates the positions

$$
\begin{array}{c}
!(N \& N) \\
q \quad q \\
n \quad m
\end{array}
$$

so that there are strategies

$$
\begin{array}{c}
N \& N \xrightarrow{\text{add}} N \\
q \quad q \\
n \quad m
\end{array}
$$

both of which compute addition of natural numbers. These strategies both implement addition, but their algorithms are slightly different, which illustrates the intensional nature of game semantics.

At this point, let us consider the game $(N \Rightarrow N) \Rightarrow N$ of higher-order functions, which is higher-order because the domain $N \Rightarrow N$ is the game of functions. Note that the domain is the exponential $!(N \Rightarrow N)$, so a strategy $\phi$ on the game $(N \Rightarrow N) \Rightarrow N$ may interact with an input strategy $f$ on $!(N \Rightarrow N)$ given by Opponent any finite number of times. Each interaction between $\phi$ and $f$ reveals an input-output pair of $f$, but this process will never collect the complete information about $f$ because there are infinitely many input-output pairs of $f$. For instance, the strategy $\text{pazo} : (N \Rightarrow N) \Rightarrow N$ that computes the sum $f(0) + f(1)$ for a given function $f : \mathbb{N} \Rightarrow \mathbb{N}$ plays by

$$
\begin{array}{c}
!(!(N[0] \rightarrow N[1])) \xrightarrow{\text{pazo}} N[2] \\
q[1] \\
0[0] \\
q[0] \\
1[0] \\
m[1] \\
n + m[2]
\end{array}
$$

This play can be read as follows:
1. Opponent’s question \(q_{[2]}\) for an output (‘What is your output?’);
2. Player’s question \(q_{[1]}\) for an input function (‘Wait, your first output please!’);
3. Opponent’s question \(q_{[0]}\) for an input (‘What is your first input then?’);
4. Player’s answer, say, 0\(_{[0]}\), to the question \(q_{[0]}\) (‘Here is my first input 0.’);
5. Opponent’s answer, say, \(n_{[1]}\), to the question \(q_{[1]}\) (‘OK, then here is my first output \(n\).’);
6. Player’s question \(q_{[1]}\) for an input function (‘Your second output please!’);
7. Opponent’s question \(q_{[0]}\) for an input (‘What is your second input then?’);
8. Player’s answer, say, 1\(_{[0]}\), to the question \(q_{[0]}\) (‘Here is my second input 1.’);
9. Opponent’s answer, say, \(m_{[1]}\), to the question \(q_{[1]}\) (‘OK, then here is my second output \(m\).’);
10. Player’s answer, say, \(n + m_{[2]}\), to the question \(q_{[2]}\) (‘Alright, my output is then \(n + m\).’).

In this play, the strategy pazo has only revealed the two input-output pairs (0, \(n\)) and (1, \(m\)) of \(f\).

Finally, let us recall the composition \(\psi \circ \phi : A \Rightarrow C\) of strategies \(\phi : A \Rightarrow B\) and \(\psi : A \Rightarrow C\). For an illustration, consider the strategies succ, double : \(N \Rightarrow N\) (n.b., succ this time is not on \(N \Rightarrow N\)):  

| \(N_{[0]}\) | succ | \(N_{[1]}\) | \(q_{[1]}\) | \(2n_{[2]}\) |
|---|---|---|---|---|
| \(q_{[0]}\) | \(m_{[0]}\) | \(m + 1_{[1]}\) |

The composition double \(\circ\) succ : \(N \Rightarrow N\) is calculated as follows. First, we have to define the promotion succ\(^\dagger\) : \(!N_{[0]}\Rightarrow !N_{[1]}\) of succ, which computes just as succ : \(!N_{[0]}\Rightarrow !N_{[1]}\) for each position of \(!N_{[0]}\Rightarrow !N_{[1]}\) occurring inside \(!N_{[0]}\Rightarrow !N_{[1]}\). A typical position played by the promotion therefore looks like

\[
\begin{array}{c}
!N_{[0]} \\
\downarrow_{\text{succ}^\dagger} \\
!N_{[1]} \\
\downarrow_{q_{[1]}} \\
q_{[0]} \\
m_{[0]} \\
m + 1_{[1]} \\
\end{array}
\]

Next, we synchronise succ\(^\dagger\) and double via the codomain \(!N_{[1]}\) of succ\(^\dagger\) and the domain \(!N_{[2]}\) of double, for which Player also plays the role of Opponent in \(!N_{[1]}\) and \(!N_{[2]}\) by copying her last moves, resulting in

\[
\begin{array}{c}
!N_{[0]} \\
\downarrow_{\text{succ}^\dagger} \\
!N_{[1]} \\
\downarrow_{\text{double}} \\
!N_{[2]} \\
\downarrow_{q_{[3]}} \\
q_{[2]} \\
q_{[0]} \\
n_{[0]} \\
n + 1_{[1]} \\
\end{array}
\]

\[
\begin{array}{c}
!N_{[2]} \\
\downarrow_{\text{double}} \\
!N_{[3]} \\
\downarrow_{q_{[3]}} \\
q_{[3]} \\
\end{array}
\]

\[
\begin{array}{c}
q_{[2]} \\
q_{[0]} \\
n_{[0]} \\
n + 1_{[1]} \\
n + 1_{[2]} \\
2 \cdot (n + 1)_{[3]} \\
\end{array}
\]
where moves made for the synchronisation are marked by the square boxes just for clarity. Importantly, it is assumed that Opponent plays on the external game \( N_0 \Rightarrow N_3 \), seeing only moves of !\( N_0 \) or !\( N_3 \).

The resulting play is to be read as follows:

1. Opponent’s question \( q_3 \) for an output in !\( N_0 \) \( \Rightarrow \) !\( N_3 \) (‘What is your output?’);
2. Player’s question \( q_2 \) by double for an input in !\( N_2 \) \( \Rightarrow \) !\( N_3 \) (‘Wait, what is your input?’);
3. \( q_2 \) in turn triggers the question \( q_1 \) for an output in !\( N_0 \) \( \Rightarrow \) !\( N_1 \) (‘What is your output?’);
4. Player’s question \( q_0 \) by succ† for an input in !\( N_0 \) \( \Rightarrow \) !\( N_1 \) (‘Wait, what is your input?’);
5. Opponent’s answer, say, \( n_0 \), to \( q_0 \) in !\( N_0 \) \( \Rightarrow \) !\( N_3 \) (‘Here is an input n.’);
6. Player’s answer \( n + 1 \) to \( q_1 \) by succ† in !\( N_0 \) \( \Rightarrow \) !\( N_1 \) (‘The output is then \( n + 1 \).’);
7. \( n + 1 \) in turn triggers the answer \( n + 1 \) to \( q_2 \) in !\( N_2 \) \( \Rightarrow \) !\( N_3 \) (‘Here is the input \( n + 1 \).’);
8. Player’s answer \( 2 \cdot (n + 1) \) to \( q_3 \) by double in !\( N_0 \) \( \Rightarrow \) !\( N_2 \) (‘The output is \( 2 \cdot (n + 1)! \).’).

Finally, we hide or delete all moves with the square boxes from the play, resulting in the strategy double•succ : \( N \Rightarrow N \) for the function \( n \mapsto 2 \cdot (n + 1) \) as expected:

\[
\begin{array}{c|c|c}
N_0 & \text{double}\cdot\text{succ} & N_3 \\
\hline
q_0 & n_0 & 2 \cdot (n + 1)_3 \\
\end{array}
\]

The category of games and strategies has games as objects, and strategies \( \phi : A \Rightarrow B \) as morphisms \( A \Rightarrow B \), and the composition of strategies just sketched forms the categorical composition.

Moreover, one can compose strategies \( \alpha : A \Rightarrow B \) and \( \phi : A \Rightarrow B \) in the same vein, obtaining the composition \( \phi \circ \alpha : B \Rightarrow B \). For instance, we have the composition double•\( n = 2n \) for all \( n \in \mathbb{N} \). Alternatively, recall the terminal game \( T \), which has no move. Hence, we have the isomorphism \( A \cong T \Rightarrow A \), and we do not distinguish strategies on \( A \) and \( T \Rightarrow A \) since they are essentially the same. As a result, the composition \( \phi \circ \alpha : B \Rightarrow B \) can be recasted as the ordinary composition \( \phi \circ \alpha : T \Rightarrow B \) of \( \alpha : T \Rightarrow A \) and \( \phi : A \Rightarrow B \).

We have seen that strategies interact with each other in a step-by-step, finitary fashion. This unique, intensional computation distinguishes game semantics from other mathematical semantics.

1.4 Martin-Löf’s universes

One can extend MLTT by a ‘types of (smaller) types’ or universe introduced by Martin-Löf [ML75]. The universe enables MLTT to expand its realm of constructive mathematics significantly. For instance, the elimination rule of the natural number (N-) type with respect to the universe generates infinitely indexed dependent types such as the type of finite lists of natural numbers by mathematical induction.

Besides, the power of the universe is greatly increased when it is combined with Martin-Löf’s well-founded tree (W-) types [ML82]. For instance, MLTT together with the universe and W-types interprets Aczel’s constructive set theory [Acz82]. Moreover, the combination of the universe and W-types offers MLTT a high proof-theoretic strength among constructive formal systems [Set93, GR94].

1.5 The problem: how to encode games for identity types by strategies

For these significant roles of the universe in MLTT and constructive mathematics, it is a natural aim to extend game semantics to the universe so that its powerful combinatorial reasoning becomes available for the study of the universe and types generated by the universe. However, it is a challenge to achieve game semantics of the universe, and it has not been established in the 30 years history of game semantics.\(^3\)

\(^3\)Blot and Laird [BL18] interpret universes, but this interpretation is by domain theory, not by game semantics.
Specifically, the challenge is how to encode games for identity (Id-) types by strategies. To see this point, recall first that the game semantics [Yam22] interprets each dependent type $\Gamma \vdash A$ type roughly by a family $A = (A(\gamma))_{\gamma: \Gamma}$ of games $A(\gamma)$ indexed by strategies $\gamma$ on the game $\Gamma$ that interprets the context $\Gamma$. Also, recall that the introduction rule of each universe $\Gamma \vdash U$ type encodes the dependent type $\Gamma \vdash A$ type by a term $\Gamma \vdash \text{En}(A) : U$ in such a way that the computation rule $\Gamma \vdash \text{El}(\text{En}(A)) = A$ type holds, where the dependent type $x : U \vdash \text{El}(x)$ type embodies the elimination rule of the universe by the substitution $\Gamma \vdash u : U \mapsto \Gamma \vdash \text{El}(u)$ type. Note that each universe is a constant dependent type, and therefore the game semantics [Yam22] should interpret it by a constant family of games, which is in turn identified by a single game in the evident way. Hence, we have to define not only a game $\mathcal{U}$ that interprets the universe $U$ but also the corresponding encoding of the family $A$ of games by a strategy $\text{En}(A)$ on the function game $\Gamma \Rightarrow \mathcal{U}$ from $\Gamma$ to $\mathcal{U}$, which interprets the introduction rule, and a family $\text{El} = (\text{El}(\mu))_{\mu: U}$ of games $\text{El}(\mu)$, which interprets the elimination rule, that satisfies $\text{El}(\text{En}(A) \bullet \gamma) = A(\gamma)$ for all $\gamma : \Gamma$, which interprets the computation rule. Recall that a strategy on the game $\Gamma \Rightarrow \mathcal{U}$ is a certain kind of an algorithm that outputs a strategy on the codomain $\mathcal{U}$ from a given input strategy on the domain $\Gamma$.

Now, let us take $A$ to be the Id-type $f : N \Rightarrow N, g : N \Rightarrow \text{Id}_{N \Rightarrow N}(f, g)$ type on the function type $N \Rightarrow N$ from $N$-type $N$ to itself. Then, the game semantics [Yam22] has to interpret the encoding term $f : N \Rightarrow N, g : N \Rightarrow \text{Id}_{N \Rightarrow N}(f, g) : U$ for this Id-type by a strategy $\text{En} : (\text{Id}_{N \Rightarrow N}) \Rightarrow U$ that satisfies $\text{El}(\text{En}(\text{Id}_{N \Rightarrow N}) \bullet (f, g)) = \text{Id}_{N \Rightarrow N}((f, g))$ for all $f, g : N \Rightarrow N$, where $\text{Id}_{N \Rightarrow N}$ is the family of games that interprets the Id-type [Yam22]. Crucially, the game $\text{Id}_{N \Rightarrow N}((f, g))$ depends on the equation $f = g$. Thus, the composition $\text{En}(\text{Id}_{N \Rightarrow N}) \bullet (f, g)$ must vary over the cases $f = g$ and $f \neq g$.

Accordingly, the strategy $\text{En}(\text{Id}_{N \Rightarrow N})$ seems to be an algorithm that decides whether the equation $f = g$ holds for all $f, g : N \Rightarrow N$, a contradiction to a well-known fact in recursion theory [RR67]. This corresponds, in game semantics, to the fact that the strategy $\text{En}(\text{Id}_{N \Rightarrow N})$ can learn about only finite input-output pairs of $f$ and $g$, so it cannot decide if the equation $f = g$ holds, as illustrated by the diagram

$$
\begin{array}{c}
(N \xrightarrow{f} N) \& (N \xrightarrow{g} N) \Rightarrow \text{En}(\text{Id}_{N \Rightarrow N}) \Rightarrow \mathcal{U} \\
\gamma \\
q \\
n \\
f(n) \\
g \\
n' \\
f(n') \\
                     \vdots \\
                     ?
\end{array}
$$

Let us see more concretely that the following naive method fails due to the problem just sketched. Let us assign a natural number $q(A)$ to the game $A$ that interprets each type $\Gamma \vdash A$ type along the inductive construction of $A$, and define a game $\mathcal{U}$ in such a way that maximal positions in $\mathcal{U}$ are of the form $q, q(A)$.

$$
\begin{array}{c}
\mathcal{U} \\
q \\
q(A)
\end{array}
$$

Intuitively, the initial element $q$ is Opponent’s question ‘What is your game?’, and the second one $q(A)$ is Player’s answer ‘My game is $A$!’ Further, let $\text{En}(A) : \Gamma \Rightarrow \mathcal{U}$ be the strategy that encodes the family $A$ of games by playing $q \mapsto q(A)$ without ever computing on the domain $\Gamma$. For this game $\mathcal{U}$, the strategy $\text{En}(\text{Id}_{N \Rightarrow N})$ would decide if $f = g$ (even without interacting with $f$ or $g$), which is clearly impossible.

$$
\begin{array}{c}
\Gamma \xrightarrow{\text{En}(A)} \mathcal{U} \\
q \\
q(A)
\end{array}
$$

$$
\begin{array}{c}
\text{En}(\text{Id}_{N \Rightarrow N}((f, g))) \\
q
\end{array}
$$
Remark. This naive method does not exploit any intrinsic feature of game semantics, but it actually works for encoding all standard dependent types except Id-types. Hence, one may say that our main contribution is game semantics of the universe that subsumes the encoding of Id-types.

1.6 Our solution: encoding without deciding

A key observation behind our solution to the problem described in §1.5 is that

The game semantics [Yam22] allows the decoding function $El$ to be uncomputable without sacrificing the algorithmic nature of strategies. In particular, the strategy $\text{En}(\text{Id}_{N \Rightarrow N}) \bullet (f, g)$ does not have to decide the equality $f = g$; it only has to encode the game $\text{Id}_{N \Rightarrow N}(f, g)$.

This leads us to the following solution. Let $\sharp(1), \sharp(0), \sharp(N), \sharp(\Pi), \sharp(\Sigma), \sharp(\text{Id}) \in \mathbb{N}$ be arbitrarily fixed pairwise distinct natural numbers. We then define the game $\mathcal{U}$ in such a way that

The strategy $\text{En}(\text{Id}_{N \Rightarrow N}) \bullet (f, g) : \mathcal{U}$ plays first by computing $q \mapsto \sharp(\text{Id})$ (indicating that it encodes an Id-type) and then, depending on the next move by Opponent, by playing as the strategy $\text{En}(N \Rightarrow N)$ (indicating that the encoded Id-type is on the type $N \Rightarrow N$) or by merely copy-catting $f$ and $g$ given by Opponent in the step-by-step fashion (indicating that the encoded Id-type is between $f$ and $g$) without necessarily detecting what $f$ or $g$ is.

The point is that this method allows the strategy $\text{En}(\text{Id}_{N \Rightarrow N})$ to encode the family $\text{Id}_{N \Rightarrow N}$ without sacrificing its algorithmic nature: The copy-cat of Opponent’s strategies $f$ and $g$ is trivially computable, while the potentially infinite plays by $\text{En}(\text{Id}_{N \Rightarrow N}) \bullet (f, g) : \mathcal{U}$ faithfully encode whether or not $f = g$.

In general, positions in $\mathcal{U}$ consist of symbols $\sharp(X)$ that encode type constructions $X$ and ordinary (i.e., not necessarily symbolic) strategies. In the following, let us sketch the definition of $\mathcal{U}$.

First, we have to encode the base cases, i.e., the games $1, 0$ and $N$ that interpret One-, Zero- and N-types, respectively, by strategies on the game $\Gamma \Rightarrow U$. For this reason, $\mathcal{U}$ subsumes the positions

| $\mathcal{U}$ | $\mathcal{U}$ | $\mathcal{U}$ |
|----------------|----------------|----------------|
| $q$ | $\sharp(1)$ | $\sharp(0)$ |
| $\sharp(N)$ | $\sharp(0)$ | $\sharp(N)$ |

so that there are strategies $\text{En}(1), \text{En}(0), \text{En}(N) : \Gamma \Rightarrow \mathcal{U}$ that compute respectively by

$$
\text{En}(1) : q \mapsto \sharp(1), \quad \text{En}(0) : q \mapsto \sharp(0), \quad \text{En}(N) : q \mapsto \sharp(N).
$$

Next, we consider the inductive step to encode Pi- and Sigma-types. Assume that a family $A = (A(\gamma))_{\gamma \in \Gamma}$ of games $A(\gamma)$ interprets a type $\Gamma \vdash A$ type, and a strategy $\text{En}(A) : \Gamma \Rightarrow \mathcal{U}$ interprets the encoding $\Gamma \vdash \text{En}(A) : U$. For simplicity, let $\Gamma$ be the empty context; thus, $\Gamma$ is the terminal game $T$ that has only the trivial strategy, and $A$ is identified with a game. Assume further that a family $B = (B(\alpha))_{\alpha \in A}$ of games $B(\alpha)$ interprets a type $x : A \vdash B$ type, and a strategy $\text{En}(B) : A \Rightarrow \mathcal{U}$ interprets the encoding $x : A \vdash \text{En}(B) : U$. Recall that the game semantics [Yam22] interprets the Pi-type $\Pi(A, B)$ type and the Sigma-type $\Sigma(A, B)$ type by (the singleton families of) the games $\Pi(A, B)$ and $\Sigma(A, B)$, respectively. Then, there must be strategies $\text{En}(\Pi(A, B)), \text{En}(\Sigma(A, B)) : T \Rightarrow \mathcal{U} \cong \mathcal{U}$ that respectively encode these families. For this reason, the game $\mathcal{U}$ also subsumes the positions

| $\mathcal{U}$ | $\mathcal{U}$ | $\mathcal{U}$ | $\mathcal{U}$ |
|---------------|---------------|---------------|---------------|
| $q$ | $\sharp(\Pi)$ | $q$ | $\sharp(\Pi)$ | $q$ | $\sharp(\Sigma)$ | $q$ | $\sharp(\Sigma)$ |
| $a_1$ | $b_1$ | $a_1$ | $b_1$ | $a_2$ | $b_2$ | $a_2$ | $b_2$ |
| $a_2$ | $b_2$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |

where $a_1, a_2, \ldots$ are moves played by the strategy $\text{En}(A) : U$, and $b_1, b_2, \ldots$ by the strategy $\text{En}(B) : A \Rightarrow U$.

In other words, we define the strategy $\text{En}(\Pi(A, B))$ to be the pairing $(\text{En}(A), \text{En}(B)) : U \& (A \Rightarrow U)$.

\footnote{This is because the game semantics of Pi-types (Definition [Yam22]) reveals the type dependency only gradually so that it is not necessary to compute the value of the function $El$ in one go. We shall come back to this point in Example 2.36.}
Theorem 1.1

\[ \text{U} \]

The universe \( \alpha \) where the moves \( U \) of these universes is lifted to the general case, where the game \( \Gamma \) can be different from the trivial one, by a code in some universe \( U \). Based on the idea just sketched, we obtain the following main results of the present work:

1.5: Having established the game \( \text{U} \), this theorem in turn extends the independence proof of the previous work [Yam22]:

\[ \text{U} \]

Finally, there must be a strategy \( \text{En}(\text{Id})(\alpha, \alpha') \) for each pair \( \alpha, \alpha' : A \) of strategies that encodes (the singleton family of) the game \( \text{Id}_A((\alpha, \alpha')) \). For this reason, we further add the positions

| \( q \) | \( \text{z}(\text{Id}) \) | \( \text{z}(\text{Id}) \) |
|---|---|---|
| \( a_1 \) | \( c_1 \) | \( c'_1 \) |
| \( a_2 \) | \( c_2 \) | \( c'_2 \) |
| \vdots | \vdots | \vdots |

where the moves \( a_1a_2 \ldots \) are played by the strategy \( \text{En} : \text{U} \), the moves \( c_1c_2 \ldots \) by the strategy \( \alpha : A \) and the moves \( c'_1c'_2 \ldots \) by the strategy \( \alpha' : A \). In other words, we define the strategy \( \text{Id}_A((\alpha, \alpha')) \) to be the pairing \( \langle \text{En}(A), (\alpha, \alpha') \rangle : \text{U} \langle A & A \rangle \) prefixed by the moves \( q \) of \( \text{Id} \). It is easy to see how this can be lifted to the general case, where the game \( \Gamma \) can be different from the trivial one \( T \), and the strategies \( \alpha, \alpha' \) are on the game \( \Pi(\Gamma, A) \) for Pi-types. This generalisation is illustrated in the next paragraph.

Now, let us see how this idea solves the problem sketched in §1.5. Instead of the trivial assumption \( \Gamma = T \), take \( \Gamma = (N \Rightarrow N) \& (N \Rightarrow N) \), and further let \( A \) be the singleton family \( \{ N \Rightarrow N \} \), together with the projections \( \alpha = \pi_1 : (N \Rightarrow N) \& (N \Rightarrow N) \rightarrow (N \Rightarrow N) \) and \( \alpha' = \pi_2 : (N \Rightarrow N) \& (N \Rightarrow N) \rightarrow (N \Rightarrow N) \). Then, we define the strategy \( \text{En}(\text{Id}_{N \Rightarrow N}((\pi_1, \pi_2))) : (N \Rightarrow N) \& (N \Rightarrow N) \rightarrow \text{U} \) in play in either of the following ways illustrated in Figure 1 depending on the moves played by Opponent.

In the first two patterns of Figure 1 the strategy \( \text{En}(\text{Id}_{N \Rightarrow N}((\pi_1, \pi_2))) \) encodes the underlying family \( A = \{ N \Rightarrow N \} \), where recall that function \( \Rightarrow \) on games is the trivial class of Pi \( \Pi \) on games. Hence, the family \( A \) is encoded simply by the pairing \( \langle \text{En}(N), \text{En}(N) \rangle : \text{U} \& \text{U} \) prefixed by the moves \( q \) of \( \Pi \).

In the last two patterns of the figure, what the strategy \( \text{En}(\text{Id}_{N \Rightarrow N}((\pi_1, \pi_2))) \) does is essentially to copy-cat the input strategies \( f \) or \( g \) given by Opponent. Hence, this strategy is trivially effective, but also its (potentially infinite) plays collectively have the complete information about \( f \) and \( g \), in particular whether or not \( f = g \). In this way, we overcome the main problem sketched in §1.5.

1.7 Lifting to the cumulative hierarchy of universes

The universe \( \text{U} \) does not have its own code since otherwise the code \( \Gamma \vdash \text{En}(U) : \text{U} \) leads to inconsistency known as Girard’s paradox [Gir72]. To address this problem, Martin-Löf excluded the judgement \( \Gamma \vdash \text{En}(U) : \text{U} \) and proposed a cumulative hierarchy of universes \( \text{U}_k \in \text{U} \). The first universe \( \text{U}_0 \) does not have its own code \( \text{En}(U_0) \), but the second universe \( \text{U}_1 \) has. Similarly, the second universe \( \text{U}_1 \) does not have its own code \( \text{En}(U_1) \), but the third universe \( \text{U}_2 \) has, and so on. The hierarchy of these universes is cumulative: If \( i < j \), then the larger universe \( \text{U}_j \) has all codes in the smaller one \( \text{U}_i \), plus the code \( \text{En}(U_i) \). In this way, the hierarchy collectively encodes every type, including the universes themselves, by a code in some universe \( \text{U}_k \). Note that the universe \( \text{U} \) is identified with the first universe \( \text{U}_0 \).

Having established the game \( \text{U} \) for the universe \( \text{U} \), it is straightforward to interpret the cumulative hierarchy \( \text{U}_k \in \text{U} \) of universes by a cumulative hierarchy \( \text{U}_k \in \text{U} \) of games: For the base case, we define \( \text{U}_0 := \text{U} \); for the inductive step, we define \( \text{U}_{k+1} \) by adding the code \( \text{En}(U_i) \) for \( i = 0, 1, \ldots, k \) to \( \text{U} \).

1.8 Main results

Based on the idea just sketched, we obtain the following main results of the present work:

Theorem 1.1 (computational game semantics of universes). The game semantics of MLTT [Yam22] is extendable to the cumulative hierarchy of universes without sacrificing its computability.

This theorem in turn extends the independence proof of the previous work [Yam22]:
Figure 1: An illustration of the strategy on the encoding of the Id-type between functions
Corollary 1.2 (independence of Markov’s principle). Markov’s principle is independent from MLTT equipped with the cumulative hierarchy of universes.

This corollary illustrates a strong advantage of game semantics: The combinatorial reasoning of game semantics such as the independence proof remains valid even when game semantics is extended to new types. Hence, when the game semantics of MLTT has been extended to other types, the meta-theoretic results on MLTT shown by the game semantics will be automatically extended to those types as well.

This advantage makes game semantics a quite powerful tool for the study of MLTT. In contrast, the syntactic proof given by Coquand and Manna [MC17], for instance, does not have such a modular property because an extension of MLTT may invalidate their syntactic, inductive reasoning.

1.9 Our contributions and related work

Our main contribution is the first game semantics of universes (Theorem 1.1) in the 30 years history of game semantics. The main challenge in achieving game semantics of universes is how to encode games by strategies, especially games that interpret Id-types (§1.5). We solve this problem by the novel idea to encode games by strategies that consist of both symbolic and non-symbolic computations (§1.6), while we allow the decoding function El to be uncomputable (without sacrificing the effective nature of the game semantics of MLTT [Yam22]). This idea in turn requires a nontrivial recursive definition of games for interpreting universes, and our main technical contribution is to establish such a definition.

Another contribution is to show the independence of Markov’s principle from MLTT equipped with the cumulative hierarchy of universes (Corollary 1.2). This result demonstrates the modular property of the game-semantic reasoning: A meta-theoretic result on MLTT given by the game semantics of MLTT is automatically extended to new types as soon as the game semantics is extended to the types.

Abramsky et al. [AJV15] establishes the first intensional semantics of a fragment of MLTT. However, they interpret Sigma-types indirectly by a list construction, not by games, which makes an interpretation of universes hopeless. Besides, their method is valid only for a specific class of types [VJA18, Figure 7], which excludes, e.g., the list type. Because the list type is constructible by the elimination rule of N-type with respect to universes, this limitation also implies that their approach cannot interpret universes.

Blot and Laird [BL18, Table 3] also interpret a universe, for which they write \( \Gamma \vdash e : \mathcal{I} \) type, but their interpretation is by domain theory [GHK+03], not game semantics. Besides, they do not interpret Id-types and instead sketch how to interpret Id-types by finite tuples of Boolean-type [BL18 §9]; however, this method does not work in the presence of N-type since the set \( \mathbb{N} \) of all natural numbers is unbounded.

Finally, Coquand and Manna [MC17] show the independence of Markov’s principle from MLTT equipped with a single universe for the first time in the literature. Their independence proof is syntactic, which stands in contrast to our game-semantic proof. As we have mentioned in §1.8, their syntactic proof is not straightforward to extend to other types, while our game-semantic proof is.

1.10 The structure of the present article

The rest of the present article proceeds as follows. We first prepare for the interpretation of universes by recalling the game semantics of MLTT [Yam22] in §2. We then proceed to our main contribution in §3 game semantics of the cumulative hierarchy of universes. We next present immediate corollaries of this result in §4 including the independence of Markov’s principle from MLTT equipped with the hierarchy of universes. We finally draw a conclusion and propose some future work in §5.

Notation. We use the following notations:

- We use bold small letters \( s, t, u, v \), etc. for sequences, in particular \( \epsilon \) for the empty sequence, and small letters \( a, b, m, n, x, y \), etc. for elements of sequences;
- We define \( \pi := \{1, 2, \ldots, n\} \) for each \( n \in \mathbb{N}_+ := \mathbb{N} \setminus \{0\} \), and \( \overline{\pi} := \emptyset \);
- We write \( x_1x_2\ldots x_{|s|} \) for a sequence \( s = (x_1, x_2, \ldots, x_{|s|}) \), where \( |s| \) is the length of \( s \), define \( s(i) := x_i \) \( (i \in [s]) \) and write \( a \in s \) if \( a = s(j) \) for some \( j \in [s] \);
- A concatenation of sequences \( s \) and \( t \) is represented by their juxtaposition \( st \) (or \( s.t \)), but we often write \( as, tb, ucv \) for \( (a)s, t(b), u(c)v \), and so on;
We write Even(s) (resp. Odd(s)) if s is of even- (resp. odd-) length, and given a set S of sequences and P ∈ {Even, Odd}, we define $S^P := \{ s \in S \mid P(s) \}$;

We write $s \preceq t$ if s is a prefix of a sequence t, and given a set S of sequences, $Pref(S)$ for the set of all prefixes of sequences in S, i.e., $Pref(S) := \{ s \mid \exists t \in S. s \preceq t \}$.

## 2 Review: game semantics of Martin-Löf type theory

In this section, we recall the game semantics of MLTT given in the previous work [Yam22]. To this end, we first recall games and strategies à la McCusker [McC98] (with the slight modifications made by the previous work [Yam22]) that interpret simple type theories [AM99] in §2.1 since the previous work is based on this variant of games and strategies. We then review basic definitions and results of the game semantics of MLTT [Yam22] in §2.2. Our exposition is minimal; see the tutorial [AM99] and the previous work [Yam22] for more explanations and examples.

### 2.1 Games and strategies

*Games* are a class of directed rooted forests. For technical convenience, we identify games with the sets of all paths from the roots, called *positions*. The vertices of games are called *moves*, and positions must be *legal*. These concepts are centered around the structure of *arenas*.

**Definition 2.1** (moves [Yam22]). Let us fix arbitrary pairwise distinct symbols O, P, Q and A, and call them *labels*. A *move* is a triple $m^{xy} := (m, x, y)$ such that $x \in \{O, P\}$ and $y \in \{Q, A\}$. We abbreviate moves $m^{xy}$ as $m$ and instead define $\lambda(m) := xy$, $\lambda^{OP}(m) := x$ and $\lambda^{QA}(m) := y$.

We call a move $m$ an *O-move* if $\lambda^{OP}(m) = O$, a *P-move* if $\lambda^{OP}(m) = P$, a *question* if $\lambda^{QA}(m) = Q$, and an *answer* if $\lambda^{QA}(m) = A$.

**Definition 2.2** (arenas [McC98 Yam22 HO00]). An *arena* is a pair $G = (M_G, \rightarrow_G)$ of

- A set $M_G$ of moves;
- A subset $\rightarrow_G$ of the cartesian product ($\{\ast\} \cup M_G \times M_G$, where $\ast$ (also written $\cdot_G$) is an arbitrarily fixed element such that $\ast \notin M_G$, called the *enabling relation*, that satisfies
  - (E1) If $\ast \rightarrow_G m$, then $\lambda(m) = OQ$;
  - (E2) If $m \rightarrow_G n$ and $\lambda^{QA}(n) = A$, then $\lambda^{QA}(m) = Q$;
  - (E3) If $m \rightarrow_G n$ and $n \neq \ast$, then $\lambda^{OP}(m) \neq \lambda^{OP}(n)$.

We call moves $m \in M_G$ initial if $\ast \rightarrow_G m$, and define the set $M_G^{\text{init}} := \{ m \in M_G \mid \ast \rightarrow_G m \}$ of all initial moves of $G$. An arena $G$ is *well-founded*, i.e., there is no sequence $(m_i)_{i \in \mathbb{N}}$ of moves $m_i \in M_G$ such that $\ast \rightarrow_G m_0$ and $m_i \rightarrow_G m_{i+1}$ for all $i \in \mathbb{N}$.

Strictly speaking, positions of games are sequences of moves equipped with *pointers*:

**Definition 2.3** (j-sequences [HO00 Cog93 McC98]). An *occurrence* in a finite sequence $s$ is a pair $(s(i), i)$ such that $i \in |s|$. A *justified (j-) sequence* is a pair $s = (s, J_s)$ of a finite sequence $s$ of moves and a map $J_s: |s| \rightarrow \{0\} \cup |s| - 1$ such that $0 \leq J_s(i) < i$ for all $i \in |s|$, called the *pointer* of the j-sequence. An occurrence $(s(i), i)$ is *initial* in $s$ if $J_s(i) = 0$.

We say that the occurrence $(s, J_s(i))$, $J_s(i)$ is the *justifier* of a non-initial one $(s(i), i)$ in $s$, and $(s(i), i)$ is justified by $(s, J_s(i), J_s(i))$ in $s$. A j-sequence $s$ is in an arena $G$ if its elements are moves of $G$, and its pointer respects the relation $\rightarrow_G$ in $G$, i.e., $\forall i \in |s|: (J_s(i) = 0 \Rightarrow s(i)) \vee (J_s(i) \neq 0 \Rightarrow s(J_s(i)) \rightarrow_G s(i))$. We write $J_G$ for the set of all j-sequences in $G$. A *justified (j-) subsequence* of a j-sequence $s$ is a j-sequence $t$, written $t \subseteq s$, such that $t$ is a subsequence of $s$, and $J_t(i) = j$ if and only if $J_s(i) = j$ for some $n \in \mathbb{N}$, with the occurrences $(s(J_s(i)), J_s(i))$ for $k = 1, 2, \ldots, n - 1$ deleted in $t$.

**Convention.** We are henceforth casual about the distinction between moves and occurrences, and by abuse of notation, we frequently keep the pointer $J_s$ of each j-sequence $s = (s, J_s)$ implicit since it is mostly obvious, and abbreviate occurrences $(s(i), i)$ in $s$ as $s(i)$. We write $J_s(s(i)) = J_s(j)$ if $J_s(i) = j > 0$. 12
Definition 2.4 (views \texttt{[Coq95, HO00, McC98]}). The \textit{P-view} $[s]$ and the \textit{O-view} $[s]$ of a $j$-sequence $s$ are the $j$-subsequences of $s$ defined by the induction

- $[\epsilon] := \epsilon$;
- $[sm] := [s].m$ if $m$ is a P-move;
- $[sm] := m$ if $m$ is initial;
- $[smt n] := [s].mn$ if $n$ is an O-move such that $m$ justifies $n$;
- $[\epsilon] := \epsilon$;
- $[sm] := [s].m$ if $m$ is an O-move;
- $[smt n] := [s].mn$ if $n$ is a P-move such that $m$ justifies $n$.

Definition 2.5 (legal positions \texttt{[AM99, McC98]}). A \textit{legal} position is a $j$-sequence $s$ such that

- (\textit{Alternation}) If $s = s_1 m n s_2$, then $\lambda_{OP}(m) \neq \lambda_{OP}(n)$;
- (\textit{Visibility}) If $s = t m u$ with $m$ non-initial, then $J_s(m)$ occurs in the P-view $[t]$ if $m$ is a P-move, and in the O-view $[t]$ otherwise.

A legal position is \textit{in an arena} $G$ if it is a $j$-sequence in $G$ (Definition 2.3). We write $\mathcal{L}_G$ for the set of all legal positions in $G$.

Definition 2.6 (games \texttt{[McC98, AM99, Yam22]}). A \textit{game} is a set $G$ of legal positions such that

1. $G$ is nonempty and \textit{prefix-closed} (i.e., $sm \in G \Rightarrow s \in G$);
2. $\text{Arn}(G) := (M_G, \top_G)$ is an arena, where $M_G := \{ s(i) \mid s \in G, i \in \overline{s} \}$ and $\top_G := \{ (s, s(j)) \mid s \in G, J_s(j) = 0 \} \cup \{ (s(i), s(j)) \mid s \in S, J_s(j) = i > 0 \}$.

A game $G$ is \textit{well-founded} if so is the arena $\text{Arn}(G)$, and \textit{well-opened} if each of its elements has at most one initial occurrence (i.e., the conjunction of $sm \in G$ and $m \in M_G$ implies $s = \epsilon$). We call elements of $G$ (valid) positions in $G$. A \textit{subgame} of $G$ is a game $H \subseteq G$, and $\text{sub}(G) := \{ H \mid H \text{ is a subgame of } G \}$.

Example 2.7. The simplest game is the terminal game $T := \{ \epsilon \}$ which only has the trivial position $\epsilon$. The flat game on a given set $S$ is the game flat$(S) := \text{Pref}(\{ q^{0Q} m^{PA} \mid m \in S \})$, where $q$ is an arbitrarily fixed element such that $q \notin S$, and $q^{0Q}$ justifies $m^{PA}$. Consider, for instance, the empty game $0 := \text{flat}(\emptyset)$ and the natural number game $N := \text{flat}(\mathbb{N})$. As the notation indicates, the empty game 0 interprets Zero-type, and the natural number game $N$ interprets N-type (2.4).

On the other hand, \textit{strategies} on a game $G$ are algorithms for Player about how to play on $G$:

Definition 2.8 (strategies \texttt{[McC98]}). A \textit{strategy} on a game $G$ is a subset $\sigma \subseteq G^{\text{Even}}$, written $\sigma : G$, that is nonempty, \textit{even-prefix-closed} (i.e., $sm \in \sigma \Rightarrow s \in \sigma$) and \textit{deterministic} (i.e., $smn, smn' \in \sigma \Rightarrow smn = smn'$). We write $\text{st}(G)$ for the set $\{ \sigma \mid \sigma : G \}$ of all strategies on $G$.

The idea is that a strategy $\sigma : G$ describes for Player how to play on the game $G$ by the computation $sm \in G^{\text{Odd}} \Rightarrow smn \in \sigma$ (u.b., $m$ is an O-move, and $n$ is a P-move), if any, which is \textit{deterministic} by the determinacy of $\sigma$, and in general \textit{partial} since there can be no output $smn \in \sigma$.

Example 2.9. The terminal game $T$ only has the trivial strategy $\top := \{ \epsilon \}$, and the flat game $\text{flat}(S)$ on a given set $S$ has strategies $\bot := \{ \epsilon \}$ and $\omega := \{ \epsilon, q m \}$ for each $m \in S$.

Strategies are \textit{unrestricted} computations, e.g., they can be \textit{partial}, some of which do not correspond to proofs in logic or formal systems. This motivates \textit{winning} and \textit{well-bracketing} on strategies: Winning strategies correspond to proofs in classical logic, and winning, well-bracketed ones to proofs in intuitionistic logic. Because the underlying logic of MLTT is intuitionistic, we achieve a tight correspondence between MLTT game semantics by focusing on winning, well-bracketed strategies.

Definition 2.10 (constraints on strategies \texttt{[Coq95, Lm97, McC98, AM99]}). A strategy $\sigma : G$ is
• Total if it always responds: \( \forall s \in \sigma, \exists m \in G \Rightarrow \exists m'n \in \sigma; \)

• Innocent if it only depends on P-views: \( \forall s, t \in G, [s] = [t] \Rightarrow [s]t \in \sigma, [s'm'] = [t]; \)

• Noetherian if there is no strictly increasing (with respect to the prefix relation \( \preceq \)) infinite sequence of elements in the set \( [\sigma] := \{ [s] \mid s \in \sigma \} \) of all P-views in \( \sigma; \)

• Winning if it is total, innocent and noetherian;

• Well-bracketed if its ‘question-answering’ in P-views is in the ‘last-question-first-answered’ fashion: If \( sqt a \in \sigma, \) where \( \lambda N \mathcal{Q}(a) = Q, \lambda N \mathcal{A}(a) = A \) and \( J(sqt a) = q, \) then each question occurring in \( t', \)

where the P-view \([sqt]\) has \([sqt]\) if \( t' \) by visibility, justifies an answer occurring in \( t'. \)

Example 2.11. The strategies \( \top : T \) and \( \bot : N \) for all \( n \in \mathbb{N} \) are winning and well-bracketed, while the strategies \( \bot : 0 \) and \( \bot : N \) are not even total, let alone winning.

Let us next recall standard constructions on games and strategies.

Convention. For brevity and readability, we omit ‘tags’ for disjoint union \( \oplus \). For instance, we write \( x \in A \oplus B \) if \( x \in A \) or \( x \in B \); also, given relations \( R_A \subseteq A \times A \) and \( R_B \subseteq B \times B \), we write \( R_A \oplus R_B \) for the relation on \( A \oplus B \) such that \( (x, y) \in R_A \oplus R_B :\Rightarrow (x, y) \in R_A \lor (x, y) \in R_B \).

Definition 2.12 (constructions on arenas \cite{McC98}). Given arenas \( A \) and \( B \), we define

\( A \cup B := (M_A \cup M_B, r_A \cup r_B); \)

\( A \twoheadrightarrow B := (\{ (a^{(x)}y) \mid a^{xy} \in M_A \} \cup M_B, r_A \twoheadrightarrow B) \), \( O^\bot := P, \) \( P^\bot := O, \) \( \star \twoheadrightarrow B m := \star \twoheadrightarrow_B m \) and \( m \twoheadrightarrow_{A \cup B} n := m \twoheadrightarrow A n \lor m \twoheadrightarrow B n \lor (\star \twoheadrightarrow_B m \land \star \twoheadrightarrow_A n) \).

Definition 2.13 (constructions on games \cite{McC98}). Given games \( G \) and \( H \), we define

• The tensor \( G \otimes H := \{ s \in \mathcal{L}(G) \otimes \mathcal{L}(H) \mid \forall X \in \{ G, H \}, s \mid X \subseteq X \} \) of \( G \) and \( H \), where \( s \mid X \subseteq s \) consists of occurrences of moves in \( X \);

• The exponential \( !G := \{ s \in \mathcal{L}(G) \mid \forall i \in [s], J_s(i) = 0 \Rightarrow s \mid \{(s(i), i)\} \subseteq G \} \) of \( G \), where \( s \mid \{(s(i), i)\} \subseteq s \) consists of occurrence in \( s \) hereditarily justified by the initial one \( (s(i), i) \) in \( s \);

• The product \( G \& H := \{ s \in \mathcal{L}(G) \otimes \mathcal{L}(H) \mid (s \mid G \in G \land s \mid H = e) \lor (s \mid G = e \land s \mid H \in H) \} \) of \( G \) and \( H \);

• The linear implication \( G \rightarrow_H H := \{ s \in \mathcal{L}(G) \rightarrow_H \mathcal{L}(H) \mid s \mid G \in G, s \mid H \in H \} \) from \( G \) to \( H \),

also written \( H^G \), where \( s \mid G \in G \) is obtained from \( s \mid G \) by modifying all the moves \( m(a^{xy}) \) occurring in \( s \) into \( G \) into \( m^{xy} \);

• The implication \( G \Rightarrow_H H := !G \rightarrow_H H \) from \( G \) to \( H \).

Notationally, exponential \( ! \) precedes other constructions on games, while tensor \( \otimes \) and product \( \& \) do linear implication \( \rightarrow \) and implication \( \Rightarrow \).

Definition 2.14 (constructions on strategies \cite{McC98}). Given strategies \( \phi : A \rightarrow B, \sigma : C \rightarrow D, \tau : A \rightarrow C, \psi : B \rightarrow C \) and \( \theta : A \rightarrow B \), we define

• The copy-cat \( \text{cp}_A := \{ s \in (A) \rightarrow (A)^{\text{Even}} \mid \forall t \leq s. \text{Even}(t) \Rightarrow t \mid A^{\perp}_0 = t \mid A^{\perp}_0 \} \) on \( A; \)

• The derection \( \text{der}_A := \{ s \in (A \rightarrow A)^{\text{Even}} \mid \forall t \leq s. \text{Even}(t) \Rightarrow t \mid A^{\perp} = t \mid A \} \) on \( A; \)

• The tensor \( \phi \otimes \sigma := \{ s \in A \otimes C \rightarrow B \otimes D \mid s \mid A, B \in \phi, s \mid C, D \in \sigma \} \) of \( \phi \) and \( \sigma \), where \( s \mid A, B \subseteq s \) (resp. \( s \mid C, D \subseteq s \) ) consists of occurrences of moves in \( A \) or \( B \) (resp. \( C \) or \( D \));

• The pairing \( \langle \phi, \tau \rangle := \{ s \in A \rightarrow B \otimes C \mid (s \mid A, B \in \phi \land s \mid C = e) \lor (s \mid A, C \in \tau \land s \mid B = e) \} \) of \( \phi \) and \( \tau; \)
• The composition $\phi;\psi := \{ s \mid A, C \mid s \in \phi \parallel \psi \}$ of $\phi$ and $\psi$ (n.b., $\phi;\psi$ is also written $\psi \circ \phi$), where $\phi \parallel \psi := \{ s \in \mathcal{J} \mid s \in A, B \mid s \in \phi, s \parallel B \mid s \in C \parallel \psi, s \parallel B \}$, $\mathcal{J} := \mathcal{J}_{\text{Ann}((A \rightarrow B) \rightarrow (B \rightarrow C))} \cup \{ s \mid B_{1}, B_{2} \}$ is obtained from $s \mid B_{0}, B_{1}$ by applying the operation $\langle \cdot \rangle_{\parallel} : m^{x+y} \mapsto m^{x+y}$ (Definition 2.12) on all moves $m^{x+y}$;

• The promotion $\theta^{\parallel} := \{ s \mid (A \rightarrow !B)^{\text{Even}} \mid \forall i \in |s|, \mathcal{J}_{s}(i) = 0 \Rightarrow s \mid \{ (s(i), i) \} \in \theta \}$ of the strategy $\theta$.

Example 2.15. The promotion $\text{suc}^{\parallel} : !N \rightarrow !N$ of the strategy $\text{suc} := \{ q_{0}, q_{1}, n_{0}, n + 1 \mid n \in \mathbb{N} \} : N_{[0]} \Rightarrow N_{[1]}$
computes as sketched in the introduction (§1.3).

Let us summarise the present section by:

Definition 2.16 (categories of games [McC98, Yam22]). The category $G_{I}$ consists of

• Well-opened games as objects;
• Strategies on implication $A \Rightarrow B$ as morphisms $A \rightarrow B$;
• The composition $\psi \bullet \phi := \psi \circ \phi^{\parallel} : A \Rightarrow C$ of strategies as the composition of morphisms $\phi : A \rightarrow B$ and $\psi : B \rightarrow C$;
• The dereliction $\text{der}_{A}$ as the identity on each object $A$.

The subcategory $\forall G_{I}$ (resp. $\forall G$) of $G_{I}$ consists of well-founded, well-opened games as objects, and winning (resp. winning, well-bracketed) strategies as morphisms.

We have to focus on well-opened games in these categories since otherwise the identities would not be well-defined [McC98, pp. 42–43]. We use the subscript $(\_)^{\parallel}$ in order to distinguish these categories from the linear ones, in which morphisms $A \rightarrow B$ are strategies on the linear implication $A \rightarrow B$.

Notation. We are not bothered about the distinction between strategies on games $G$ and $T \Rightarrow G$.

2.2 Game semantics of Martin-Löf type theory

The previous work [Yam22] establishes game semantics of MLTT based on games and strategies recalled in the previous section. The central idea of the previous work is to generalise games into predicate (p-) games, which corresponds to the generalisation of simple types to dependent types:

Definition 2.17 (p-games [Yam22]). A predicate (p-) game is a pair $\Gamma = (|\Gamma|, |\Gamma|)$ of a game $|\Gamma|$ and a family $|\Gamma| = (|\Gamma|(\gamma))_{\gamma \in \gamma}$ of subgames $|\Gamma|(\gamma) \subseteq |\Gamma|$. It is well-founded (resp. well-opened) if so is $|\Gamma|$.

Example 2.18. Given a game $G$, we have the p-game $G := (G, \kappa_{G})$, where $\kappa_{G}$ is the constant family at $G$. Clearly, $G$ and $\mathcal{P}(G)$ are essentially the same. We abbreviate $\mathcal{P}(T)$, $\mathcal{P}(0)$ and $\mathcal{P}(N)$ as $T$, $0$ and $N$, and call them the terminal p-game, the empty p-game and the natural number p-game, respectively.

Before discussing strategies on p-games, we need a few preliminary concepts:

Definition 2.19 (liveness ordering [Chu00]). The liveness ordering is a partial order $\preceq$ between games $\mathcal{C}$ (Definition 8 and Theorem 9], which defines $G \preceq H$ to mean that $O$ (resp. $P$) is less (resp. more) restricted in $G$ than in $H$, i.e., they satisfy

1. If $s \in (G \cap H)^{\text{Even}}$ and $sm \in H^{\text{Odd}}$, then $sm \in G^{\text{Odd}}$;
2. If $tl \in (G \cap H)^{\text{Odd}}$ and $tlr \in G^{\text{Even}}$, then $tlr \in H^{\text{Even}}$.

Definition 2.20 (closures of strategies [Yam22]). The closure of a strategy $\sigma : G$ with respect to another game $H$ is the subgame $\overline{\sigma}_{H} := \{ e \} \cup \{ sm \mid s \in H^{\text{Odd}} \mid s \in \overline{\sigma}_{H} \} \cup \{ tl \mid s \in \sigma \mid tl \in \overline{\sigma}_{H} \} \subseteq \sigma \cup H$.

We see by induction that $\overline{\sigma}_{G} = \sigma \cup \{ sm \in G \mid s \in \sigma \}$ holds for all strategies $\sigma : G$. Moreover:
Proposition 2.21 (liveness characterisation [Yam22]). Assume \( \sigma : G \) and \( H \in \text{sub}(G) \).

1. \( \overline{\sigma}_H^\text{Even} : H \) if and only if \( \overline{\sigma}_G \not\preceq H \);
2. If \( \overline{\sigma}_G \preceq H \), then \( \overline{\sigma}_H^\text{Even} = \sigma \cap H \).

This proposition enables us to define strategies on p-games as follows:

Definition 2.22 (strategies on p-games [Yam22]). A strategy on a p-game \( \Gamma \), written \( \gamma : \Gamma \), is a strategy \( \gamma : |\Gamma| \) such that \( \overline{\sigma}_{|\Gamma|} \not\preceq \Gamma(\gamma) \). It is total (resp. innocent, noetherian, well-bracketed) if so is \( \gamma \cap \Gamma(\gamma) = \Gamma(\gamma) \).

We write \( \text{st}(\Gamma) \) for the set \( \{ \gamma : \Gamma \} \) of all strategies on a p-game \( \Gamma \) and define \( \overline{\sigma}_\Gamma := \overline{\sigma}_{\Gamma(\gamma)} \) for all \( \gamma : \Gamma \). A position in \( \Gamma \) is a prefix of a sequence \( q_1 \gamma s \) such that \( \gamma : \Gamma \) and \( s \in \overline{\sigma}_\Gamma \), where \( q_1 \) is an arbitrarily fixed element such that \( q_1 \not\in M_{|\Gamma|} \), \( q_1 \gamma \) is called an initial protocol, and \( s \) is called an actual position.

A play in \( \Gamma \) proceeds as follows. First, \( \text{Judge} \) asks Player a question \( q_1 \) ("What is your strategy?") and she answers it by a strategy \( \gamma : \Gamma \) ("It is \( \gamma!'\)). After this initial protocol, an ordinary play on the game \( \Gamma(\gamma) \) between Player and Opponent follows, in which Player must use the declared one \( \gamma \) restricted to \( \Gamma(\gamma) \), i.e., \( \gamma \cap \Gamma(\gamma) = \overline{\sigma}_{\Gamma(\gamma)} = \Gamma(\gamma) \). Thus, \( \gamma : \Gamma \) is winning (resp. well-bracketed) if so is \( \overline{\sigma}_{\Gamma(\gamma)} = \Gamma(\gamma) \).

Judge and the initial protocol are mere devices for requiring Player to judge and the initial protocol are mere devices for requiring Player to answer a question \( q_1 \). Hence, strategies on \( \Gamma \) & \( \Delta \) are the pairings \( \langle \sigma \rangle : \Gamma \) for all \( \sigma : \Gamma \); strategies on \( \text{Even} \) & \( \Delta \) are those \( \sigma : \Gamma \) which \( \sigma \) are those \( \sigma : \Gamma \) the strategy \( \sigma \) if it does not hold if the game \( \Gamma(\gamma) \) is undefined.

Definition 2.23 (product and tensor on p-games [Yam22]). The product of p-games \( \Gamma \) and \( \Delta \) is the p-game \( \Gamma \& \Delta \) defined by \( \langle \Gamma \& \Delta \rangle := \langle \Gamma \rangle \& \langle \Delta \rangle \) and \( \langle \Gamma \& \Delta \rangle(\langle \gamma \rangle, \langle \delta \rangle) := \langle \Gamma \rangle(\gamma) \& \langle \Delta \rangle(\delta) \) for all \( \langle \gamma \rangle, \langle \delta \rangle : \langle \Gamma \& \Delta \rangle \), and their tensor is the p-game \( \Gamma \otimes \Delta \) defined by \( \langle \Gamma \otimes \Delta \rangle := \langle \Gamma \rangle \otimes \langle \Delta \rangle \) and \( \langle \Gamma \otimes \Delta \rangle(\langle \sigma \rangle) := \langle \sigma \rangle \otimes \langle \Gamma \rangle \otimes \langle \delta \rangle \otimes \langle \Delta \rangle \) for all \( \langle \sigma \rangle : \langle \Gamma \& \Delta \rangle \).

Definition 2.24 (countable tensor [Yam22]). The countable tensor of a family \( (G_i)_{i \in N} \) of subgames \( G_i \subseteq H \) is the subgame \( \otimes_{i \in N} G_i := \{ s \in \text{!H} \mid \forall j \in N. s \upharpoonright j \in G_j \} \subseteq \text{!H} \).

Definition 2.25 (exponential on p-games [Yam22]). The exponential of a p-game \( \Gamma \) is the p-game \( !\Gamma \) defined by \( \langle !\Gamma \rangle := \langle !\Gamma \rangle \) and \( \langle !\Gamma \rangle(\langle \sigma \rangle) := \otimes_{i \in N} \Gamma(\langle \sigma \rangle \upharpoonright i) \) for all \( \langle \sigma \rangle : \langle !\Gamma \rangle \).

Hence, strategies on \( \Gamma \& \Delta \) are the pairings \( \langle \gamma, \delta \rangle \) of \( \gamma : \Gamma \) and \( \delta : \Delta \), strategies on \( \Gamma \otimes \Delta \) are the tensors \( \gamma \otimes \delta \) of \( \gamma : \Gamma \) and \( \delta : \Delta \), and strategies on \( !\Gamma \) are those \( \sigma : \langle !\Gamma \rangle \) such that \( \{ s \mid i \in \sigma \} : \Gamma \) for all \( i \in N \).

Definition 2.26 (categories of p-games [Yam22]). The category \( \text{PG}_\Gamma \) consists of

- Well-opened p-games as objects;
- Strategies on the implication \( \Gamma \Rightarrow \Delta \) as morphisms \( \Gamma \to \Delta \);
• The composition \( \psi \cdot \phi := \psi \circ \phi : \Gamma \to \Theta \) of strategies as the composition of morphisms \( \phi : \Gamma \to \Delta \) and \( \psi : \Delta \to \Theta \);
• The dereliction \( \text{der}_{|\Gamma|} : \Gamma \to \Gamma \) as the identity \( \text{id}_\Gamma \) on each object \( \Gamma \).

The subcategory \( \text{LPG}_\Gamma \) (resp. \( \text{WPG}_\Gamma \)) of \( \text{PG}_\Gamma \) consists of well-founded, well-opened p-games as objects, and winning (resp. winning, well-bracketed) strategies as morphisms.

Because the underlying logic of MLTT is intuitionistic, the previous work [Yam22] focuses on the category \( \text{WPG}_\Gamma \). It establishes game semantics of MLTT by showing that the category \( \text{WPG}_\Gamma \) gives rise to abstract semantics of MLTT, called a category with families (CwF) [Dyb96].

**Definition 2.27** (CwFs [Dyb96] [Hof97]). A category with families (CwF) is a tuple
\[
\mathcal{C} = (\mathcal{C}, \text{Ty}, \text{Tm}, \{\_\}, T, \_p, \_v, \langle \_\_ \rangle),
\]
where
• \( \mathcal{C} \) is a category with a terminal object \( T \in \mathcal{C} \);
• \( \text{Ty} \) assigns, to each object \( \Gamma \in \mathcal{C} \), a set \( \text{Ty}(\Gamma) \) of \( \text{types} \) in the context \( \Gamma \);
• \( \text{Tm} \) assigns, to each pair \( (\Gamma, A) \) of a context \( \Gamma \in \mathcal{C} \) and a type \( A \in \text{Ty}(\Gamma) \), a set \( \text{Tm}(\Gamma, A) \) of \( \text{terms} \) of type \( A \) in the context \( \Gamma \);
• To each morphism \( \phi : \Delta \to \Gamma \), \( \langle \_ \rangle \) assigns a map \( \langle \phi \rangle : \text{Ty}(\Gamma) \to \text{Ty}(\Delta) \), called the substitution on \( \text{types} \), and a family \( \langle \phi \rangle_{A \in \text{Ty}(\Gamma)} \) of maps \( \langle \phi \rangle_A : \text{Tm}(\Gamma, A) \to \text{Tm}(\Delta, A\{\phi\}) \), called the substitutions on \( \text{terms} \);
• \( \_p \) (resp. \( \_v \)) associates each pair \( (\Gamma, A) \) of a context \( \Gamma \in \mathcal{C} \) and a type \( A \in \text{Ty}(\Gamma) \), a context \( \Gamma.A \in \mathcal{C} \), called the comprehension of \( A \);
• \( \_p \) (resp. \( \_v \)) associates each pair \( (\Gamma, A) \) of a context \( \Gamma \in \mathcal{C} \) and a type \( A \in \text{Ty}(\Gamma) \) with a morphism \( p_A : \Gamma.A \to \Gamma \) (resp. a term \( v_A \in \text{Tm}(\Gamma.A, A\{p_A\}) \)), called the first projection on \( A \) (resp. the second projection on \( A \));
• \( \langle \_p, \_v \rangle \) assigns, to each triple \( (\phi, A, \tilde{\alpha}) \) of a morphism \( \phi : \Delta \to \Gamma \), a type \( A \in \text{Ty}(\Gamma) \) and a term \( \tilde{\alpha} \in \text{Tm}(\Delta, A\{\phi\}) \), a morphism \( \langle \phi, A, \tilde{\alpha} \rangle : \Delta \to \Gamma.A \), called the extension of \( \phi \) by \( \tilde{\alpha} \),

that satisfies, for any \( \Theta \in \mathcal{C} \), \( \varphi : \Theta \to \Delta \) and \( \alpha \in \text{Tm}(\Gamma, A) \), the equations
\[
\begin{align*}
A\{\text{id}_\Gamma\} &= A, \\
A(\phi \circ \varphi) &= A\{\phi\}\{\varphi\}, \\
\alpha\{\text{id}_\Gamma\} &= \alpha, \\
\alpha(\phi \circ \varphi) &= \alpha(\phi)(\varphi)_A, \\
p_A \circ (\phi, \tilde{\alpha})_A &= \phi, \\
v_A(\phi, \tilde{\alpha})_A &= \tilde{\alpha}, \\
(\phi, A) \circ \varphi &= (\phi \circ \varphi, \alpha(\varphi)_A), \\
(p_A, v_A)_A &= \text{id}_{\Gamma.A}.
\end{align*}
\]

We sometimes write \( T_{\mathcal{C}} \), \( \text{Term}_\mathcal{C} \) and so on when we would like to emphasise the underlying CwF \( \mathcal{C} \). Roughly, judgements of MLTT are interpreted in a CwF \( \mathcal{C} \) by

\[
\begin{align*}
\Gamma \vdash \text{ctx} \Rightarrow [\Gamma] &\in \mathcal{C} \\
\Gamma \vdash A \text{ type} \Rightarrow [A] &\in \text{Ty}(\Gamma) \\
\Gamma \vdash a : A \Rightarrow [A] &\in \text{Tm}(\Gamma, A) \\
\Gamma \vdash \alpha = \alpha' : A \Rightarrow [\alpha] = [\alpha'], \\
\end{align*}
\]

where \([\_]\) denotes the semantic map or interpretation. See [Hof97] for the details.

In the following, we recall the additional structures on the category \( \text{WPG}_\Gamma \) that lift it to a CwF. First, types in the CwF \( \text{WPG}_\Gamma \) are dependent p-games:

**Definition 2.28** (dependent p-games [Yam22]). A linearly dependent predicate \((p-)\) game over a p-game \( \Gamma \) is a pair \( L = ([L], \|L\|) \) of a game \( [L] \) and a family \( \|L\| = (L(\gamma))_{\gamma \in \text{WPG}_\Gamma(\Gamma)} \) of p-games \( L(\gamma) \) such that \( |L(\gamma)| = |L| \). It is well-opened (resp. well-founded) if so is \( |L| \). The extension of the family \( \|L\| \) is the family \( L^* = (L^*(\gamma))_{\gamma: \Gamma} \) of p-games \( L^*(\gamma) \) defined by

\[
L^*(\gamma) := \begin{cases}
L(\gamma) & \text{if } \gamma \in \text{WPG}_\Gamma(\Gamma); \\
\mathcal{P}(|L|) & \text{otherwise}.
\end{cases}
\tag{3}
\]

A dependent predicate \((p-)\) game over \( \Gamma \) is a linearly dependent one over the exponential \(!\Gamma\).
Notation. We write \( \mathcal{D}_\ell(\Gamma) \) (resp. \( \mathcal{D}_w(\Gamma) \)) for the set of all linearly dependent p-games (resp. well-founded ones) over \( \Gamma \), and \( \{\Gamma\}'_w \) or \( \{\Gamma\}'_w \) for the constant one at \( \Gamma' \), i.e., \( \{\Gamma\}'_w := (\Gamma', \gamma : \Gamma \mapsto \Gamma') \).

Let \( \mathcal{D}(\Gamma) := \mathcal{D}_w(\Gamma) \) and \( \mathcal{D}'(\Gamma) := \mathcal{D}_w(\Gamma) \). We often write \( \gamma_0^\downarrow \) for an arbitrary element of \( \mathcal{WPG}_G(\Gamma) \), where \( \gamma_0 \in \mathcal{WPG}_G(\Gamma) \), since elements of \( \mathcal{WPG}_G(\Gamma) \) are all innocent and so promotions of elements of \( \mathcal{WPG}_G(\Gamma) \) are winning, well-bracketed strategies on the following p-games:

**Definition 2.29** (linear-pi and pi [Yam22]). Let \( L \) be a linearly dependent p-game over a p-game \( \Gamma \), and \( A \) be a dependent p-game over \( \Gamma \). The linear-pi from \( \Gamma \) to \( L \) is the p-game \( \Pi_{\ell}(\Gamma, L) \) defined by \( |\Pi_{\ell}(\Gamma, L)| := |L|_{\Pi} \) and for all \( \phi \in |\Pi_{\ell}(\Gamma, L)| \)

\[
\Pi_{\ell}(\Gamma, L)\langle \phi \rangle := \{ e \} \cup \{ sm \in |\Pi_{\ell}(\Gamma, L)|_{\text{Odd}} \mid s \in |\Pi_{\ell}(\Gamma, L)\langle \phi \rangle|, \exists \gamma : \Gamma, sm \in L^*(\gamma)(\phi \circ \gamma)^{\gamma_0^\downarrow} \} \\
\cup \{ tl \in |\Pi_{\ell}(\Gamma, L)|_{\text{Even}} \mid tl \in |\Pi_{\ell}(\Gamma, L)\langle \phi \rangle|, \forall \gamma : \Gamma, tl \in L^*(\gamma)(\phi \circ \gamma)^{\gamma_0^\downarrow} \Rightarrow tl \in L^*(\gamma)(\phi \circ \gamma)^{\gamma_0^\downarrow} \},
\]

and the \( \Pi \) from \( \Gamma \) to \( A \) is the linear-pi \( \Pi(\Gamma, A) := \Pi_{\ell}(\Gamma, A) \). We write \( \Gamma \Rightarrow A \) for \( \Pi(\Gamma, A) \) if \( A \) is constant.

Finally, comprehensions in the CwF \( \mathcal{WPG} \) are given by:

**Definition 2.30** (sigma [Yam22]). The sigma of a p-game \( \Gamma \) and a dependent p-game \( A \) over \( \Gamma \) is the p-game \( \Sigma(\Gamma, A) \) defined by \( |\Sigma(\Gamma, A)| := |\Gamma| \cup |A| \) and \( \Sigma(\Gamma, A)((\gamma, \alpha)) := (\gamma) \cup A^*(\gamma^\downarrow)(\alpha) \) for all \( (\gamma, \alpha) : |\Sigma(\Gamma, A)| \). We write \( \Gamma \Rightarrow A \) for \( \Sigma(\Gamma, A) \) if \( A \) is constant.

We are now ready to recall:

**Theorem 2.31** (a game-semantic CwF [Yam22]). The category \( \mathcal{WPG}_G \) gives rise to a CwF as follows:

- The terminal p-game \( T \in \mathcal{WPG}_G \) forms a terminal object;
- We define \( \text{Ty}(\Gamma) := \mathcal{D}_w(\Gamma) \) and \( \text{Ty}(\Gamma) := \mathcal{WPG}_G(\Pi(\Gamma, A)) \) (\( A \in \mathcal{D}_w(\Gamma) \));
- Given a morphism \( \phi : \Delta \to \Gamma \), we define \( \Sigma_{\Delta}(\phi) : \text{Ty}(\Gamma) \to \text{Ty}(\Delta) \) by \( |A| \subseteq \Sigma_{\Delta}(\phi) := |A| \cup A[\phi] \) for all \( A \in \text{Ty}(\Gamma) \) and \( A[\phi] \in \mathcal{WPG}_G(\Delta) \), and define \( \varphi_A : \text{Ty}(\Gamma, A) \to \text{Ty}(\Delta, A[\phi]) \) by \( \varphi_A(\alpha) := \alpha \cdot \phi \) for all \( \alpha \in \text{Ty}(\Delta, A[\phi]) \);
- We define \( \Lambda_A : \Sigma(\Gamma, A), p_A := \text{der}_{|\Gamma|} : \Sigma(\Gamma, A) \to \Gamma, v_A := \text{der}_{|A|} : \Sigma(\Gamma, A), A[p_A] \) and \( (\phi, \alpha)_A : (\phi, \alpha) : \Delta \to \Sigma(\Gamma, A) \).

Given \( \Gamma \in \mathcal{WPG}_G \) and \( A \in \mathcal{D}_w(\Gamma) \), we write \( \mathcal{WPG}_G(\Gamma, A) \) for the set \( \text{Ty}(\Gamma, A) \) of all terms. We often omit subscripts on components of \( \mathcal{WPG}_G \) when they are evident.

Strictly speaking, a CwF only interprets the core part of MLTT common to all types. For interpreting One-, Zero-, N-, Pi-, Sigma- and Id-types, we need to equip the CwF \( \mathcal{WPG}_G \) with semantic type formers [Hof97]. That is, these type formers that interpret these types. In the following, we only sketch the game-semantic type formers on the CwF \( \mathcal{WPG}_G \), leaving the general definition of semantic type formers to Hofmann [Hof97]. Let us fix an objects \( \Delta, \Gamma \in \mathcal{WPG}_G \) and types \( A \in \mathcal{D}_w(\Gamma) \) and \( B \in \mathcal{D}_w(\Sigma(\Gamma, A)) \).

**Theorem 2.32** (game semantics of Pi-types [Yam22]). \( \mathcal{WPG}_G \) strictly supports Pi-types, where

- \( \text{(Pi-FORM)} \) A dependent p-game \( \Pi(\Gamma, A, B) \in \mathcal{D}_w(\Gamma) \) is given by \( |\Pi(\Gamma, A, B)| := |A| \Rightarrow |B| \) and \( \Pi(\Gamma, A, B)((\gamma_0^\downarrow)) := \Pi(\Delta(\gamma_0^\downarrow), B_{\gamma_0}) \) for each \( \gamma_0 \in \mathcal{WPG}_G(\Gamma) \), and another dependent p-game \( B_{\gamma_0} \in \mathcal{D}_w(\Delta(\gamma_0)) \) by \( |B_{\gamma_0}| := |B| \) and \( B_{\gamma_0}(\alpha_0^\downarrow) := B(\gamma_0, \alpha_0^\downarrow) \) for each \( \alpha_0 \in \mathcal{WPG}_G(\Delta(\gamma_0^\downarrow)) \). We write \( A \Rightarrow B \) for \( \Pi(\Gamma, A, B) \) if \( B_{\gamma_0} \) is constant for each \( \gamma_0 \in \mathcal{WPG}_G(\Gamma) \). Note that the equation

\[
\Pi(\Gamma, A, B)(\phi) := \Pi(A, B(\phi_0))
\]

holds for each morphism \( \phi : \Delta \to \Gamma \) \( \text{(Pi-SUBST)} \), where \( \phi_0^\downarrow := (\phi \cdot p, v) : \Delta.A(\phi) \to \Gamma.A \).

- \( \text{(Pi-INTRO)} \) Given a term \( \beta \in \mathcal{WPG}_G(\Sigma(\Gamma, A), B) \), another term \( \lambda_{A,B}(\beta) \in \mathcal{WPG}_G(\Gamma, \Pi(\Gamma, A, B)) \) is obtained from \( \beta \) by adjusting tags or currying \( \beta \) with respect to the adjunction between tensor \( \odot \) and linear implication \( \Rightarrow \) (thanks to the evident isomorphism \( |\Sigma(\Gamma, A)| = !|\Gamma| \otimes !|A| \)). We often omit the subscripts \( (\phi)_{A,B} \) on \( \lambda_{A,B} \) and the inverse \( \lambda_{A,B}^{-1} \).
Theorem 2.33 (game semantics of Sigma-types \cite{Yam22}). WPG\(_{\Gamma}\) strictly supports Sigma-types, where

- \((\Sigma-\text{FORM})\) Similarly to Pi-types, we define \(\Sigma(A, B) := ([A] \& [B], \langle\Sigma(A(\gamma_0^0), B(\gamma_0^1))\rangle)\) \(\gamma_0^0 \in \text{WPG}_{\Gamma}(\Pi(A, B))\) and \(\alpha \in \text{WPG}_{\Gamma}(\Pi(A, B))\). We often omit the subscripts \(\langle\cdot\rangle_{A,B}\) on \(\text{App}_{A,B}\).

Theorem 2.34 (game semantics of atomic types \cite{Yam22}). \text{WPG}_{\Gamma}\ supports One-, Zero- and N-types, where their formation rules are given by constant dependent p-games at the terminal p-game \(\top\), the empty p-game 0 and the natural number p-game \(N\), for which we write 1, 0 and \(N\), respectively.

Theorem 2.35 (game semantics of Id-types \cite{Yam22}). \text{WPG}_{\Gamma}\ supports Id-types, where

- \((\text{Id-\text{FORM}})\) Let \(T' := \text{flat}\{\{\}\}\) (Example 2.31), where \(\sqrt{\_}\) is any element. We define a dependent p-game \(\text{Id}_A \in \mathcal{G}^p(\Sigma(\Gamma, A), A^+)\) by \(|\text{Id}_A| := T' \wedge \text{Id}_A(\langle\gamma_0, \alpha_0, \alpha_0^0\rangle) := \begin{cases} (T', \kappa_T) & \text{if } \alpha_0 = \alpha_0^0, \\ (T', \kappa_0) & \text{otherwise,} \end{cases}\) for all \(\langle\gamma_0, \alpha_0, \alpha_0^0\rangle \in \text{WPG}_{\Gamma}(\Sigma(\Gamma, A), A^+)\), where \(\kappa_X\) is the constant family at a game \(X\).

- \((\text{Id-\text{INTRO}})\) Let \(\text{Ref}_{\text{Id}} := \langle\sqrt{\_}, \text{refl}_{\text{Id}}\rangle \in \text{WPG}_{\Gamma}(\Sigma(\Gamma, A), \Sigma(\Sigma(\Gamma, A), A^+), \text{Id}_A)\), where \(\text{refl}_{\text{Id}} \in \text{WPG}_{\Gamma}(\Sigma(\Gamma, A), \text{Id}_A(\langle A\rangle))\) is \(\sqrt{\_}\) (Example 2.31) up to tags.

Example 2.36. Consider the interpretation of the Id-type \(f : N \Rightarrow N, g : N \Rightarrow N \vdash \text{Id}_{N \Rightarrow N}(f, g)\) type in the CwF \text{WPG}_{\Gamma} \cite{Yam22}, which is the p-game \(\Pi((N \Rightarrow N) \& (N \Rightarrow N), \text{Id}_{N \Rightarrow N}(\pi_1, \pi_2))\). The component of the codomain \(\text{Id}_{N \Rightarrow N}(\pi_1, \pi_2)\) is in general not decidable because any play in this p-game can observe only finite information about two input strategies on the domain. Nevertheless, this is not a problem because the codomain component of each \(\pi\) (Definition 2.29) is specified only gradually (and often incompletely) along the gradual (and often incomplete) disclosure of input strategies on the domain by Opponent. Accordingly, assuming a p-game \(\mathcal{U}\) for the universe, a strategy \(\text{En}(\text{Id}_{N \Rightarrow N}(\pi_1, \pi_2)) : (N \Rightarrow N) \& (N \Rightarrow N), \text{Id}_{N \Rightarrow N}(\pi_1, \pi_2))\), if any, only has to encode the currently possible components of the codomain \(\text{Id}_{N \Rightarrow N}(\pi_1, \pi_2)\) at each moment; it does not have to decide if the two input strategies on the domain are equal. We emphasise that this intensionality is highly nontrivial, and it distinguishes game semantics from other semantics of MLTT such as domains and realisability \cite{Pa03, Str01, BL15}. Moreover, this observation is the starting point of our solution to the main problem \(\S 3\) in achieving game semantics of the universe sketched in \S 1.6.

3 Game semantics of universes

This section presents our main contribution: game semantics of the cumulative hierarchy of universes.

To this end, we first recall the semantic type former for the cumulative hierarchy of universes:

Definition 3.1 (categorical semantics of universes \cite{Hof97}). A CwF \(\mathcal{C}\) supports universes if

- \((\text{U-\text{FORM}})\) Given an object \(\Gamma \in \mathcal{C}\), there is a type \(U^k_{\mathcal{C}}(\Gamma)\) for each natural number \(k \in \mathbb{N}\), called the \((k + 1)st\) universe in the context \(\Gamma\), where we often omit the superscript \(\langle\cdot\rangle_{\mathcal{C}}(\Gamma)\) (when the object \(\Gamma\) is obvious) and/or the subscript \(\langle\cdot\rangle_k\) (when the index \(k\) is unimportant);

- \((\text{U-\text{INTRO}})\) Given a type \(A \in \text{Ty}(\Gamma)\), there is a term \(\text{En}_k(A) \in \text{Ty}_k(\Gamma)\) for some \(k \in \mathbb{N}\), subsuming \(\text{En}_k(U^k_{\mathcal{C}}(\Gamma))\) \(\in \text{Ty}_k(\Gamma, U^k_{\mathcal{C}}(\Gamma))\) for each \(k \in \mathbb{N}\), where we often omit the subscript \(\langle\cdot\rangle_k\) on \(\text{En}\);

- \((\text{U-\text{ELIM}})\) Each term \(\psi \in \text{Ty}_k(\Gamma)\) induces a type \(\text{El}_k(\psi) \in \text{Ty}(\Gamma)\), where we often omit the subscript \(\langle\cdot\rangle_k\) on \(\text{El}\);
• (U-COMP) El(En(A)) = A;
• (U-CUMUL) If ψ ∈ Tm(Γ, U_1), then ψ ∈ Tm(Γ, U_{k+1});
• (U-SUBST) U_k^{[\Delta]} = U_k^{[\Delta]} ∈ Ty(\Delta) for each morphism ϕ : Δ → Γ;
• (En-SUBST) En(A) = En[A(\phi)] ∈ Ty(∆, U).

The axiom U-CUMUL requires the hierarchy (U_k)_{k∈N} of universes U_k to be cumulative. For achieving game semantics of the cumulative hierarchy of universes, it suffices to equip our game-semantic CwF WPG with this semantic type former because then the semantic type former will automatically induce game semantics of the cumulative hierarchy of universes as described in Hofmann [Ho97].

3.1 Universe predicate games

For convenience, we employ the following reformulation Id'(α, α') of Id-types that satisfy the axiom Id'-SUBST corresponding to Id-SUBST. In fact, the type Id_A ∈ Ty(Γ, A.A^+) is equivalent to the family (Id_A'(α, α'))_{α, α'∈Tm(Γ, A)} of types Id_A'(α, α') ∈ Ty(Γ): The former is recovered from the latter by

\[ Id_A := Id_A'(v{p}, v), \]

and the latter from the former by

\[ Id_A'(α, α') := Id_A'{⟨⟨ idr, α⟩, α'⟩}. \]

We also note that the axiom Id-SUBST implies the equation

\[ Id_A'(α, α')φ = Id_A'{⟨⟨ idr, α⟩, α'⟩φ} \]

\[ = Id_A'{⟨⟨ ϕ, αφ, α'φ⟩⟩} \]

\[ = Id_A'{⟨⟨ ϕ o p, v o p, v⟩⟩} \]

\[ = Id_A'{⟨⟨ idr, αφ⟩, α'φ⟩⟩} \]

\[ = Id_A'(φ) \]

for each terms α, α' ∈ Tm(Γ, A) and morphism φ : Δ → Γ, which we call the axiom Id'-SUBST. Conversely, the axiom Id'-SUBST implies the axiom Id-SUBST because

\[ Id_A φ_{A,A^+} = Id_A'(v{p}, v){ϕ p, v} \]

\[ = Id_A(φ) \]

\[ = Id_A'(φ) \]

This in particular implies the equation

\[ Id_A'(α, α')γ_0 = Id_A'(α, α' • γ_0) \]

for all γ_0 ∈ WPG. We leave it to the reader to reformulate the other axioms on Id-types in such a way that they correspond to this reformulation. From now on, we simply write Id(α, α') for Id'(α, α').

Then, as sketched in §1.6, the main idea for the construction of our game-semantic type former for the cumulative hierarchy of universes is centred around the following universe p-games:

**Definition 3.2** (universe p-games). Let us fix an injection z_0 : {1, 0, N, Π, Σ, Id} → N. For each natural number k ∈ N, the (k + 1)st universe predicate (p-) game is the constant p-game U_k on the game U_k := ∪_{j∈N} U_k^{(j)} together with an arbitrarily fixed injection z_k : {1, 0, N, Π, Σ, Id} ∪ {U_j | j < k} → N that conservatively extends z_{k−1}, where U_k^{(j)} is a p-game inductively defined as follows:
1. **(Base case)** We define the p-game

\[ U_k^{(0)} := \mathcal{P}(\text{Pref}(\{ q^{OQ}_k(Y)^P | Y \in \{ 0, 1, N, U_j, j < k \} )) \), \]

where \( q^{OQ} \) justifies \( \sharp_k(Y)^P \), together with a function

\[ E_k^{(0)} : \text{WP}_{G1}(U_k^{(0)}) \rightarrow \text{ob}(\text{WP}_{G1}) \]

\[ \sharp_k(X)^\uparrow \mapsto X. \]

Abusing notation, we lift this function to a dependent p-game \( E_k^{(0)} \in \mathcal{D}(U_k^{(0)}) \) by

\[ |E_k^{(0)}| := \bigcup_{\sharp_k(X)^\uparrow \in \text{WP}_{G1}(U_k^{(0)})} E_k^{(0)}(\sharp_k(X)^\uparrow) \]

\[ \|E_k^{(0)}\| : \sharp_k(X)^\uparrow \mapsto E_k^{(0)}(\sharp_k(X)^\uparrow), \]

where recall Example [24] for the notation \( \sharp_k(X) \). We also write \( U_k^{(0)} \) for the constant p-game \( \{ U_k^{(0)} \} \).

Moreover, for each object \( \Gamma \in \text{WP}_{G1} \), we further lift this dependent p-game to a function

\[ E_k^{(0)} : \text{WP}_{G1}(\Gamma, U_k^{(0)}) \rightarrow \mathcal{D}(\Gamma) \]

\[ \psi \mapsto E_k^{(0)}(\psi), \]

where the dependent p-game \( E_k^{(0)}(\psi) \in \mathcal{D}(\Gamma) \) is given by

\[ |E_k^{(0)}(\psi)| := \bigcup_{\gamma_0^\uparrow \in \text{WP}_{G1}(\Gamma)} E_k^{(0)}(\psi \cdot \gamma_0^\uparrow) \]

\[ \|E_k^{(0)}(\psi)\| : \gamma_0^\uparrow \mapsto E_k^{(0)}(\psi \cdot \gamma_0). \]

This function \( E_k^{(0)}(\psi) \) generalises the dependent p-game \( E_k^{(0)} \) due to the evident isomorphism \( E_k^{(0)} \cong E_k^{(0)} \). We usually omit the subscript \( (\_r) \) on the function \( E_k^{(0)}(\_r) \) when it does not bring confusion.

2. **(Inductive step)** We define the p-game

\[ U_k^{(i+1)} := \mathcal{D}(U_k^{(i)}) \cup \text{Pref}(\{ q^{OQ}_k(Y)^P, s | Y \in \{ \Pi, \Sigma \}, s \in \Sigma(U_k^{(i)}), E_k^{(i)} \Rightarrow U_k^{(i)} \}) \]

\[ \cup \text{Pref}(\{ q^{OQ}_k(\text{Id})^P, t | t \in \Sigma(U_k^{(i)}), E_k^{(i)} \Rightarrow E_k^{(i)} \}) \), \]

where \( q^{OQ} \) justifies both \( \sharp(Y)^P \) and \( \sharp(\text{Id})^P \), and in turn the latter two moves justify the initial moves in \( s \) and \( t \), respectively, together with a function

\[ E_k^{(i+1)} : \text{WP}_{G1}(U_k^{(i+1)}) \rightarrow \text{ob}(\text{WP}_{G1}) \]

\[ \sharp_k(X)^\uparrow \mapsto X \]

\[ q, \sharp_k(Y)^\uparrow, (\mu, \psi)^\uparrow \mapsto Y(E_k^{(i)}(\mu), E_k^{(i)}(\psi)) \]

\[ q, \sharp_k(\text{Id})^\uparrow, (\mu, (\alpha, \alpha'))^\uparrow \mapsto \text{Id}_{E_k^{(i)}(\mu)}(E_k^{(i)}(\alpha), E_k^{(i)}(\alpha')), \]

where \( q, a, \sigma := \text{Pref}(\{ q^{OQ}u^P | u \in \sigma \})^{\text{even}} \) for each question \( q^{OQ} \), answer \( a^P \), and strategy \( \sigma \), and \( q^{OQ} \) justifies \( a^P \) and \( E_k^{(i)} \), and \( a^P \) justifies initial moves occurring in \( \sigma \).

Again, we lift this function \( E_k^{(i+1)} \) to a dependent p-game \( E_k^{(i+1)} \in \mathcal{D}(U_k^{(i+1)}) \) and further to a function \( \text{WP}_{G1}(\Gamma, U_k^{(i+1)}) \rightarrow \mathcal{D}(\Gamma) \) for each \( \Gamma \in \text{WP}_{G1} \) in the same way as the case of \( E_k^{(0)} \). We also write \( U_k^{(i+1)} \) for the constant p-game \( \{ U_k^{(i+1)} \} \) and apply the notations for \( E_k^{(0)} \) to \( E_k^{(i+1)} \).

Given an object \( \Gamma \in \text{WP}_{G1} \), we write \( U_k^{(\Gamma)} \in \mathcal{D}(\Gamma) \) for the constant dependent p-game at \( U_k \) and we often omit the superscript \( (\_r) \) on \( U_k^{(\Gamma)} \) when it does not bring confusion.

We finally define the injection

\[ \sharp := \bigcup_{k \in \mathbb{N}} \sharp_k : \{ 1, 0, N, \Pi, \Sigma, \text{Id} \} \cup \{ \mathcal{U}_j | j \in \mathbb{N} \} \rightarrow \mathbb{N}. \]
Let us emphasise that the inductive step in Definition 3.2 properly implements our idea on how to encode game semantics of Pi-, Sigma- and Id-types by strategies on games (§1.6) by nontrivial recursion. Specifically, our key technique is to define each universe p-game \( U_k \) inductively in terms of the games \( U_k^{(i)} \) (\( i \in \mathbb{N} \)) along with the construction of the function \( El_k^{(i)} \). This is the highlight of the present work.

### 3.2 Computational game semantics of the cumulative hierarchy of universes

We need one more preparation for our game semantics of universes as follows. The axiom U-INTRO (Definition 3.3) requires that every type \( A \) has its encoding \( El(A) \). As already indicated in §1.6 however, we define the encoding function \( En \) inductively along the construction of types. Accordingly, we have to restrict types in the CwF \( \text{WPG} \) to those freely generated by the type constructions, leading to:

**Definition 3.3** (a subCwF \( \text{UPG}_i \)). Let \( \text{UPG}_i \vdash \text{WPG}_i \) be the substructural CwF of \( \text{WPG}_i \) such that

- The underlying category \( \text{UPG}_i \) is the category \( \text{WPG}_i \);
- The types of \( \text{UPG}_i \) are inductively constructed from the atomic dependent p-games \( 1, 0, N \) and \( U_k \) for all \( k \in \mathbb{N} \) by the constructions \( \Pi, \Sigma, \text{Id} \);
- The types of \( \text{UPG}_i \) are given by \( \text{Tm}_{\text{UPG}_i}(\Gamma, A) := \text{Tm}_{\text{WPG}_i}(\Gamma, A) \) for all \( \Gamma \in \text{UPG}_i \) and \( A \in \text{Ty}_{\text{UPG}_i}(\Gamma) \).

**Corollary 3.4** (well-defined \( \text{UPG}_i \)). The structure \( \text{UPG}_i \) forms a well-defined CwF that supports One-, Zero-, N-, Pi-, Sigma- and Id-types in the same way as the CwF \( \text{WPG}_i \).

Proof. This corollary immediately follows from Theorem 2.31 (where the only nontrivial point is the closure of types \( \text{UPG}_i \) under substitution, but it is easily shown by induction on the types).

In addition, this CwF \( \text{UPG}_i \) also supports the cumulative hierarchy of universes:

**Theorem 3.5** (game semantics of universes). The CwF \( \text{UPG}_i \) supports universes.

Proof. Let \( \Delta, \Gamma \in \text{UPG}_i, A \in \text{Tm}_{\text{UPG}_i}(\Gamma) \) and \( \phi \in \text{UPG}_i(\Delta, \Gamma) \).

- (U-FORM) We have \( U_k^{[\Gamma]} \in \mathcal{D}(\Gamma) \) for each natural number \( k \in \mathbb{N} \) (Definition 3.2).
- (U-INTRO) Because \( A \) is constructed inductively, we can define a term \( En(A) \in \text{Tm}_{\text{UPG}_i}(\Gamma, U_k(A)) \) for some natural number \( k(A) \in \mathbb{N} \) inductively along the construction of \( A \) as follows:
  1. If \( A = 1, 0 \) or \( N \), then \( En(A) := A \in \text{Tm}_{\text{UPG}_i}(\Gamma, U_0) \);
  2. If \( A = U_i \) for some natural number \( i \in \mathbb{N} \), then \( En(U_i) := U_i \in \text{Tm}_{\text{UPG}_i}(\Gamma, U_{i+1}) \);
  3. If \( A = Y(B, C) \), where \( Y \) is \( \Pi \) or \( \Sigma \), then \( En(Y(B, C)) := q^{\mathcal{O}}(\zeta(Y))^{\text{PA}},(\lambda \in \text{En}(C)) \in \text{Tm}_{\text{UPG}_i}(\Gamma, U_{\max(k(B),k(C))}) \);
  4. If \( A = \text{Id}(\delta, \delta') \), then \( En(\text{Id}(\delta, \delta')) := q^{\mathcal{O}}(\zeta(\text{Id}))^{\text{PA}},(\text{En}(D)) \in \text{Tm}_{\text{UPG}_i}(\Gamma, U_{k(D)}) \).
- (U-ELIM) We define the function \( El_k : \text{Tm}_{\text{UPG}_i}(\Gamma, U_k^{[\Gamma]}) \cong \text{WPG}_i(\Gamma, U_k) \rightarrow \mathcal{D}(\Gamma) \) to be the union \( El_k := (\bigcup_{i \in \mathbb{N}} El_k^{(i)} : \text{WPG}_i(\Gamma, U_k) \rightarrow \mathcal{D}(\Gamma) \)
up to the isomorphism \( \text{Tm}_{\text{UPG}_i}(\Gamma, U_k^{[\Gamma]}) \cong \text{WPG}_i(\Gamma, U_k) \), where the function \( El_k^{(i)} : \text{WPG}_i(\Gamma, U_k^{(i)}) \rightarrow \mathcal{D}(\Gamma) \) is given in Definition 3.2. Note that \( El_k^{(i)}(\psi) \in \mathcal{D}(\Gamma) \) for each \( \psi \in \text{Tm}_{\text{UPG}_i}(\Gamma, U_k^{(i)}) \) is given by

\[
|El_k(\psi)| := \bigcup_{\gamma_0 \in \text{Tm}_{\text{UPG}_i}(\Gamma)} El_k(\psi \bullet \gamma_0) \quad \|El_k(\psi)\| : \gamma_0 \mapsto El_k(\psi \bullet \gamma_0).
\]
• (U-COMP) We see that the equation $\text{El}(\text{En}(A)) = A$ holds by induction on $A$, where we focus on the cases of $A = \Pi(B, C)$ and $A = \text{Id}_{D}(\delta, \delta')$ since the other cases are similar or trivial.

1. Assume $A = \Pi(B, C)$. The dependent p-game

$$\text{El} \circ \text{En}(\Pi(B, C)) = \text{El}(q \sharp (\Pi \cdot \text{En}(B), \lambda \circ \text{En}(C)))$$

consists of the underlying p-game

$$|\text{El} \circ \text{En}(\Pi(B, C))| = |\text{El}(q \sharp (\Pi \cdot \text{En}(B), \lambda \circ \text{En}(C)))|$$

$$\bigcup_{\gamma_{0} \in \text{UPG}_{1}(\Gamma)} |\Pi(\text{El}(\text{En}(B) \bullet \gamma_{0}), \text{El}(\lambda \circ \text{En}(C) \bullet \gamma_{0}))|$$

$$= \bigcup_{\gamma_{0} \in \text{UPG}_{1}(\Gamma)} (|\text{El}(\text{En}(B) \bullet \gamma_{0})| \Rightarrow |\text{El}(\text{En}(B) \bullet \gamma_{0})|)$$

$$= \bigcup_{\gamma_{0} \in \text{UPG}_{1}(\Gamma)} (|\text{El}(\text{En}(B) \bullet \gamma_{0})| \Rightarrow \bigcup_{\gamma_{0} \in \text{UPG}_{1}(\Gamma)} |\text{El}(\text{En}(B) \bullet \gamma_{0})|)$$

$$= |\Pi \circ \text{En}(B)| \Rightarrow |\Pi \circ \text{En}(C)|$$

(by the induction hypothesis)

$$= |\Pi(B, C)|$$

and the function

$$||\text{El} \circ \text{En}(\Pi(B, C))|| : \gamma_{0} \in \text{UPG}_{1}(\Gamma) \mapsto \text{El}(q \sharp (\Pi \cdot \text{En}(B), \lambda \circ \text{En}(C) \bullet \gamma_{0}))$$

$$= \Pi(\text{El}(\text{En}(B) \bullet \gamma_{0}), \text{El}(\lambda \circ \text{En}(C) \bullet \gamma_{0}))$$

$$= \Pi(\text{El}(\text{En}(B) \bullet \gamma_{0}), \text{El}(\lambda \circ \text{En}(C) \bullet \gamma_{0}))$$

(by the induction hypothesis)

$$= \Pi(B(\gamma_{0}), C_{\gamma_{0}})$$

Hence, we have shown the equation

$$\text{El} \circ \text{En}(\Pi(B, C)) = \Pi(B, C).$$

2. Assume $A = \text{Id}_{D}(\delta, \delta')$. The dependent p-game

$$\text{El} \circ \text{En}(\text{Id}_{D}(\delta, \delta')) = \text{El}(q \sharp (\text{Id}, \langle\text{En}(D), (\delta, \delta')\rangle))$$

consists of the underlying p-game

$$|\text{El} \circ \text{En}(\text{Id}_{D}(\delta, \delta'))| = |\text{El}(q \sharp (\text{Id}, \langle\text{En}(D), (\delta, \delta')\rangle))|$$

$$\bigcup_{\gamma_{0} \in \text{UPG}_{1}(\Gamma)} |\text{Id}_{\text{El}(\text{En}(D) \bullet \gamma_{0})}(\delta \bullet \gamma_{0}, \delta' \bullet \gamma_{0})|$$

$$\bigcup_{\gamma_{0} \in \text{UPG}_{1}(\Gamma)} |\text{Id}_{\text{El}(\text{En}(D) \bullet \gamma_{0})}(\delta \bullet \gamma_{0}, \delta' \bullet \gamma_{0})|$$

$$= \text{Id}_{\text{El}(\text{En}(D) \bullet \gamma_{0})}(\delta \bullet \gamma_{0}, \delta' \bullet \gamma_{0})$$

$$= \text{Id}_{\text{El}(\text{En}(D) \bullet \gamma_{0})}(\delta \bullet \gamma_{0}, \delta' \bullet \gamma_{0})$$

(by the induction hypothesis)

$$= \text{Id}_{\text{El}(\text{En}(D) \bullet \gamma_{0})}(\gamma_{0})$$

(by the equation 5).

Hence, we have shown the equation

$$\text{El} \circ \text{En}(\text{Id}_{D}(\delta, \delta')) = \text{Id}_{D}(\delta, \delta').$$
(U-CUMUL) By construction, \( \psi \in \text{Tm}_{\text{UPG}}(\Gamma, U_k) \) implies \( \psi \in \text{Tm}_{\text{UPG}}(\Gamma, U_{k+1}) \).

(U-SUBST) By construction, the equation \( U_k^{|\Gamma|}\{\phi\} = U_k^{|\Delta|} \in \text{Ty}(\Delta) \) holds.

(EN-SUBST) We see that the equation \( \text{En}(A)\{\phi\} = \text{En}(A(\phi)) \in \text{Tm}(\Delta, U) \) holds by induction on \( A \), where again we focus on the cases of \( A = \Pi(B, C) \) and \( A = \text{Id}_D(\delta, \delta') \).

1. Assume \( A = \Pi(B, C) \). We have the equation

\[
\text{En}(\Pi(B, C))\{\phi\} = q.z(\Pi)\cdot (\text{En}(B) \bullet \phi, \lambda \circ \text{En}(C) \bullet \phi)
\]

\[= q.z(\Pi)\cdot (\text{En}(B)\{\phi\}, \lambda \circ \text{En}(C(\phi_B^+))) \quad (\text{by the induction hypothesis})
\]

\[= \text{En}(\Pi(B\{\phi\}, C(\phi_B^+)))
\]

\[= \text{En}(\Pi(B, C)\{\phi\}) \quad (\text{by the equation A}).
\]

2. Assume \( A = \text{Id}_D(\delta, \delta') \). We have the equation

\[
\text{En}(\text{Id}_D(\delta, \delta'))\{\phi\} = q.z(\text{Id})\cdot (\text{En}(D) \bullet \phi, \{\delta \bullet \phi, \delta' \bullet \phi\})
\]

\[= q.z(\text{Id})\cdot (\text{En}(D(\phi)), \{\delta(\phi), \delta'(\phi)\}) \quad (\text{by the induction hypothesis})
\]

\[= \text{En}(\text{Id}_{D(\phi)})(\delta(\phi), \delta'(\phi))
\]

\[= \text{En}(\text{Id}_D(\delta, \delta')(\phi)) \quad (\text{by Id'-SUBST}).
\]

We have verified all the required axioms, completing the proof. \( \square \)

**Example 3.6.** Let us consider the interpretation of the encoding

\[ f : N \Rightarrow N, g : N \Rightarrow N \vdash \text{En}_0(\text{Id}_{N \Rightarrow N}(f, g)) : U_0 \]

of the Id-type discussed in §1.3. The strategy

\[ \psi := \text{En}_0(\text{Id}_{N \Rightarrow N}(\pi_1, \pi_2)) \vdash (N \Rightarrow N) \& (N \Rightarrow N) \rightarrow U_0 \]

that interprets this encoding of the Id-type plays as in Figure 2.

**Example 3.7.** The elimination rule of N-type with respect to a universe generates the encodings of transfinite dependent types. For instance, the encoding of the type \( x : N \vdash \text{List}_N(x) \) type of finite lists of natural numbers, which satisfies the judgemental equalities \( \text{List}_N(0) \equiv 1 \) and \( \text{List}_N(n + 1) \equiv \text{List}_N(n) \times N \), is defined by applying the elimination rule of N-type to the terms

\[ \vdash \text{En}(1) : U \quad \text{and} \quad x : N, y : U \vdash \text{En}(E l(\pi_2 y) \times N) : U. \]

Then, the strategy

\[ \psi' := \overrightarrow{N}(\text{En}(1), \text{En}(E l(\pi_2) \& N)) : N \rightarrow U_0 \]

that interprets this encoding of the list type plays as in Figure 3.

Let us note that this list type is out of the scope of the denotational semantics by Abramsky et al. [AJV15, VJA18], let alone its encoding, because their interpretation is limited to finite inductive types [VJA18, Figure 7]; also see [Yam22, §4.3] on this point. This argument in particular implies that their approach cannot interpret the combination of universes and N-type.

4 Corollaries

This last section presents corollaries of Theorem 3.3 established in the previous section. The first corollary is the effectivity of the game semantics of universes (§1.1), the second one is the independence of the axiom of equality reflection (§1.2), and the last one is the independence of Markov’s principle (§4.3).
Figure 2: The strategy on the encoding of the Id-type between functions
Figure 3: The strategy on the encoding of the list type
4.1 Effectivity of game semantics

Let us first show the effectivity of our interpretation of universes. Note that strategies in the CwF UPG\textsubscript{1} are the conventional ones (\S 2.1), which are winning and well-bracketed. Note also that much more unrestricted strategies that interpret terms in the higher-order functional programming language PCF [Sco93, Plo77] are all effective or recursive; see [AJM00, \S 5] and [HO00, \S 5.6] for the details. In essence, terms and morphisms in UPG\textsubscript{1} are winning, well-bracketed strategies in the game semantics of PCF that satisfy the additional condition imposed by p-games (Definition 2.22).

The definition of recursive strategies is therefore directly applicable to terms and morphisms in UPG\textsubscript{1}. Roughly, assuming that moves in games are encodable by natural numbers, a strategy is recursive if its computational steps are all computable (with respect to the encoding of moves by natural numbers) in the standard sense of recursion theory [RR67]. We then define:

**Definition 4.1 (an effective subCwF UPG\textsubscript{1}eff).** Let UPG\textsubscript{1}eff \hookrightarrow UPG\textsubscript{1} be the lluf substructural CwF of the CwF UPG\textsubscript{1} whose terms and morphisms are all recursive.

Because strategies in UPG\textsubscript{1} that interpret terms in MLTT are much more restricted than those that interpret terms in PCF, it is just straightforward\footnote{Again, the point here is that our strategies are just the conventional ones, so the arguments of the existing methods such as [AJM00, \S 5] and [HO00, \S 5.6] are directly applicable.} to verify:

**Corollary 4.2 (effective game semantics of universes).** The CwF UPG\textsubscript{1}eff is well-defined and supports One-, Zero-, N-, Pi-, Sigma- and Id-types as well as the cumulative hierarchy of universes in the same way as UPG\textsubscript{1}. This in particular establishes effective game semantics of universes.

This corollary implies that our game semantics of MLTT equipped with the aforementioned types only employs recursive strategies, i.e., the game semantics is computational. Because universes are types of types or sets of sets, this computational result is nontrivial.

4.2 Independence of equality reflection

Next, let us show the independence of the axiom of equality reflection [Pal98] from MLTT: Given terms \(\psi, \psi' \in \text{Tm}(\Gamma, C)\), if El(\(\psi\)) = El(\(\psi'\)) \in Ty(\(\Gamma\)), then \(\psi = \psi'\). Then, a key observation is that, by the intensional nature of our game semantics, there can be more than one term that encodes the same type. For instance, the term \(\text{En}(1) \in \text{Tm}(T.N, U)\) that encodes One-type \(1 \in \text{Ty}(T.N)\) plays by

\[
\begin{array}{c}
T.N \xrightarrow{\text{En}(1)} U \\
\downarrow q \\
\sharp(1)
\end{array}
\]

while another term \(\psi \in \text{Tm}(T.N, U)\) that plays by

\[
\begin{array}{c}
T.N \xrightarrow{\psi} U \\
\downarrow q \\
\downarrow n \\
\sharp(1)
\end{array}
\]

for all \(n \in \mathbb{N}\) also encodes the same type (n.b., this term is given by the elimination rule of N-type).

This argument together with Theorem 3.5 immediately implies:

**Corollary 4.3 (independence of equality reflection).** The axiom of equality reflection is independent from MLTT equipped with One-, Zero-, N-, Pi-, Sigma- and Id-types as well as universes.

We have seen that the intensional nature of strategies plays a crucial role for this corollary, but it is not available for other computational semantics such as domains and realizability [Pal93, Str12, BL18].
4.3 Independence of Markov’s principle

Finally, the previous work [Yam22, §4.7] shows that Markov’s principle [Mar62] is invalid in the game semantics, which implies that the principle is independent from MLTT equipped with One-, Zero-, N-, Pi-, Sigma- and Id-types. Markov’s principle is a well-known principle in constructive mathematics, and it depends on the school of constructive mathematics whether the principle is to be regarded as constructive. Roughly, the principle postulates that if it is impossible that there is no natural number \( n \in \mathbb{N} \) such that \( f(n) = 0 \) for a function \( f : \mathbb{N} \to \mathbb{N} \), then there is a natural number \( n' \in \mathbb{N} \) such that \( f(n') = 0 \).

The proof of this independence result given in the previous work is also valid for the present game semantics without any modification. This immediately extends the independence result to universes:

**Corollary 4.4** (independence of Markov’s principle from universes). Markov’s principle is independent from MLTT equipped with One-, Zero-, N-, Pi-, Sigma- and Id-types as well as universes.

Again, this game-semantic proof [Yam22] takes advantages of the intensional nature of game semantics, which is not available for other computational semantics of MLTT.

Coquand and Manna [MC17] show the independence of Markov’s principle from MLTT equipped with a single universe for the first time in the literature. Their independence proof is syntactic, which stands in contrast to our game-semantic proof. As we have mentioned, their syntactic proof is not automatically extendable to other types, and an extension can be nontrivial. In contrast, our game-semantic reasoning is modular: A meta-theoretic result on MLTT given by our game semantics is automatically extended to new types as soon as the game semantics is extended to the types. This is one of the strong advantages of the game-semantic approach for the study of type theory and constructive mathematics.

5 Conclusion and future work

We have established computational game semantics of the cumulative hierarchy of universes for the first time in the literature. We have also applied this game semantics to the meta-theoretic study of MLTT and shown that equality reflection and Markov’s principle are both independent from MLTT equipped with the hierarchy of universes, illustrating advantages of the game-semantic approach.

For future work, we plan to extend the game semantics further to Martin-Löf’s well-founded tree (W-) types [ML82]. The resulting game semantics will be a very powerful semantic foundation of constructive mathematics, e.g., it will interpret Aczel’s constructive set theory (CZF) [Acz86] since CZF is translatable into MLTT equipped with universes and W-types.

References

[A+97] Samson Abramsky et al., *Semantics of interaction: An introduction to game semantics*, Semantics and Logics of Computation 14 (1997), 1–31.

[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.

[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.

[AM99] Samson Abramsky and Guy McCusker, *Game semantics*, Computational Logic: Proceedings of the 1997 Marktoberdorf Summer School (Berlin, Heidelberg), Springer, 1999, pp. 1–55.

[BL18] Valentin Blot and Jim Laird, *Extensional and intensional semantic universes: A denotational model of dependent types*, Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, ACM, 2018, pp. 95–104.
An Intuitionistic Theory of Types, Twenty-five years of constructive type theory 36 (1998), 127–172.

Erik Palmgren, An information system interpretation of martin-löf’s partial type theory with universes, Information and Computation 106 (1993), no. 1, 26–60.

On universes in type theory, Twenty five years of constructive type theory (1998), 191–204.

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

Hartley Rogers and H Rogers, Theory of recursive functions and effective computability, vol. 5, McGraw-Hill, New York, 1967.

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

Anton Setzer, Proof theoretical strength of martin-löf type theory with w-type and one universe, Ph.D. thesis, Uitgever niet vastgesteld, 1993.

Joseph R Shoenfield, Mathematical logic, vol. 21, Addison-Wesley, Reading, 1967.

Thomas Streicher, Semantics of Type Theory: Correctness, Completeness and Independence Results, Springer Science & Business Media, 2012.

Anne Sjerp Troelstra and Dirk van Dalen, Constructivism in mathematics. two volumes, NorthHolland, Amsterdam (1988).

Matthijs Vákár, Radha Jagadeesan, and Samson Abramsky, Game semantics for dependent types, Information and Computation 261 (2018), 401–431.

Norihiro Yamada, A game-semantic model of computation, Research in the Mathematical Sciences 6 (2019), no. 1, 3.

Norihiro Yamada, Game semantics of martin-löf type theory, Mathematical Structures in Computer Science, to appear (2022).

Ernst Zermelo, Untersuchungen über die grundlagen der mengenlehre. i, Mathematische Annalen 65 (1908), no. 2, 261–281.