Game-theoretic Interpretation of Intuitionistic Type Theory

Norihiro Yamada

norihiro.yamada@cs.ox.ac.uk
Department of Computer Science
University of Oxford
May 11, 2016

Abstract

We present a game semantics for intuitionistic type theory. Specifically, we propose categories with families of a new variant of games and strategies for both extensional and intensional variants of the type theory with $\prod$, $\Sigma$, and Id-types as well as universes. Our games and strategies generalize the existing notion of games and strategies and achieve an interpretation of dependent types and the hierarchy of universes in an intuitive manner. We believe that it is a significant step towards a computational and intensional interpretation of the type theory.

Contents

1 Introduction 2
2 Preliminaries 5
2.1 Games and Strategies . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6
   2.1.1 Arenas . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6
   2.1.2 Justified Sequences . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6
   2.1.3 Views, Legal Positions, and Threads . . . . . . . . . . . . . . . . . . . . . . . 7
   2.1.4 Games . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8
   2.1.5 Constructions on Games . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
   2.1.6 Strategies . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 11
   2.1.7 Constructions on Strategies . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
3 Predicative Games 15
   3.1 Typed Games . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
   3.2 1-predicative and 2-predicative Games . . . . . . . . . . . . . . . . . . . . . . . 16
   3.3 Predicative Games and Generalized Strategies . . . . . . . . . . . . . . . . . . . 19
   3.4 Subgame Relation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22
   3.5 Universe Games . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22
1 Introduction

In the present paper, we propose a new game semantics for intuitionistic type theory. Our motivation is to provide a computational and intensional interpretation of the type theory for its mathematical (or semantic) justification and investigation.

*Intuitionistic type theory* (or *Martin-Löf type theory*) is an extension of the simply-typed $\lambda$-calculus that, under the Curry-Howard isomorphism, corresponds to intuitionistic predicate logic, in which types can depend on terms. Thus, it is a functional programming language as well as a formal logical system. It was proposed in 1970s\(^1\) by Martin-Löf \cite{ML82,ML84,ML98,RS84} as a foundation of constructive mathematics. Since then, several other dependent type theories\(^2\) such as Calculus of Construction \cite{CH88} were also proposed. Some of these type theories were implemented as proof assistants such as Nuprl \cite{CAB+86}, Coq \cite{T+12}, and Agda \cite{Nor07}. Based on a homotopy-theoretic interpretation, an extension of intuitionistic type theory, called *homotopy type theory* (HoTT), was recently proposed, providing new insights and having potential to be a powerful and practical foundation of mathematics \cite{Uni13}. However, a

\(^1\)Later, Martin-Löf proposed several modifications a few times.

\(^2\)Intuitionistic type theory is an instance of a dependent type theory, which refers to a type theory whose types can depend on terms.
computational interpretation of univalence axiom (UA) and higher inductive types (HIT), the core axiom and construction of HoTT, has been missing, though a significant step towards this goal was recently taken in [BCH14].

Game semantics refers to a particular kind of semantics of logics and programming languages in which types and terms are interpreted as “games” and “strategies”, respectively. Historically, game semantics gave the first syntax-independent characterization of the language PCF [AJM00, HO00, Nic94], since then, a variety of games and strategies have been proposed to characterize various programming features [McC98, HY97, La97, AM97, AHM98, MT11, AM98]. One of its distinguishing features is to interpret syntax as dynamic interactions between two players, providing a computational explanation of proofs and programs in an intuitive yet mathematically precise manner. Remarkably, game semantics for dependent type theory was not addressed until Abramsky et al. recently constructed such a model in [AJV15]. However, it does not interpret universes, an important component of intuitionistic type theory.

In this paper, we propose a new game semantics for intuitionistic type theory. Concretely, we first propose a category with families (CwF) [Dyb96, Hof97] (a categorical model of the type theory) $\mathcal{E}PG$ of a new variant of games and strategies that supports $\Pi$, $\Sigma$, and Id-types as well as universes, which induces an interpretation of the extensional variant of intuitionistic type theory (ETT). And based on $\mathcal{E}PG$, we construct another CwF $\mathcal{I}PG$ to interpret the intensional variant (ITT). Our game-theoretic models have an advantage over categorical models such as [See84, Car86, Dyb96] in their concreteness and over set-theoretic models in their intensionality. Also, our models may have advantages over realizability models [Reu99, Coq98]. Models that take (codes of) Turing machines as realizers are “too intensional” to interpret functional languages, but game-theoretic models in general have an appropriate degree of intensionality as various definability (and full abstraction) results in the literature have demonstrated; approaches that take logical calculi as realizers are syntactic models, but games and strategies are different from syntax, providing new insights and tools to analyze the type theory. When compared to the game semantics in [AJV15], our model in $\mathcal{I}PG$ is simpler and provides a more intuitive interpretation. Moreover, it interprets the hierarchy of universes (perhaps for the first time in the literature of game semantics). Furthermore, since strategies can be seen as algorithms, our interpretation is conceptually closer to the BHK-interpretation of intuitionistic logic, on which Martin-Löf type theory is based, than the homotopy-theoretic interpretation. On the other hand, our model in $\mathcal{I}PG$ refutes UA; thus, it is a future work to refine the model to capture the phenomena of identity types in HoTT. Also, it remains to establish a definability result.

The rest of the paper is structured as follows. We first review the existing game semantics in Section 2 on which we define new games and strategies for an interpretation of the type theory in Section 3 introducing particular games to interpret universes in Section 3.5. Next, defining the category of the games and strategies in Section 4 we construct a model of ETT in Section 5 in which specific types such as $\Pi$, $\Sigma$, Id-types as well as universes are interpreted. Based on the extensional model, we then present a model of ITT in Section 6 and analyze its property in Section 7. Finally, we make a conclusion and propose some future works in Section 8.

Remark. Because we shall present a model of the type theory by constructing an instance of a CwF, and its interpretation of the type theory is well-known (see [Dyb96, Hof97]), we will not

---

3 They are a “mathematician” (to prove theorems) and a “rebutter” in the case of logics, and a “computer” and “environment” in the case of programming languages.

4 For instance, a term $f : A \to B$ is interpreted as a (set-theoretic) function in a set-theoretic model, while as a strategy (which can be seen as an algorithm) in our models.

5 E.g., given a Turing machine $T_0$, we may construct a pairwise distinct sequence $T_0, T_1, T_2, \ldots$ of Turing machines by defining $T_i$, for all $i \in \mathbb{N}$, to be the Turing machine that first moves $i$-cells to the right and back to the initial position, and then computes as $T_0$. 
review the syntax of the type theory. For references of the syntax, see [Hof97, NPS90].
2 Preliminaries

In this preliminary section, we fix notation and quickly review the variant of games and strategies defined in [McC98].

Notation. We shall use the following notations throughout the present paper.

- We use bold letters $s, t, u, v, w$, etc. to denote sequences.
- We use letters $a, b, c, d, e, m, n, p, q, x, y, z$, etc. to denote elements of sequences.
- A concatenation of sequences is represented by a juxtaposition of them.
- We usually write $as, tb, uv$ for sequences $(a)s, t(b), u(c)v$, respectively.
- For readability, we sometimes write $st$ for the concatenation $st$ of sequences $s$ and $t$.
- We write even$(s)$ and odd$(t)$ to mean that sequences $s$ and $t$ are of even-length and odd-length, respectively.

- For a set $S$ of sequences, we define $S_{\text{even}} \overset{\text{df}}{=} \{ s \in S \mid \text{even}(s) \}$, $S_{\text{odd}} \overset{\text{df}}{=} \{ t \in S \mid \text{odd}(t) \}$.
- We write $s \preceq t$ (resp. $s < t$) if $s$ is a (resp. strict) prefix of a sequence $t$.
- We write $s \sqsubseteq t$ (resp. $s \sqsubset t$) if $s$ is a (resp. strict) subsequence of a sequence $t$.
- For a set $S$ of sequences, $\text{pref}(S)$ denotes the set of prefixes of sequences in $S$.
- For a partially ordered set $P$ and a subset $S \subseteq P$, we write $\sup(S)$ and $\inf(S)$ for the supremum and infimum of $S$, respectively.
- We write $\mathbb{N}$ and $\mathbb{Z}$ for the set of natural numbers and the set of integers, respectively. Moreover, for each $n \in \mathbb{Z}$, we write $\mathbb{Z}_{\geq n}$ for the set of integers $\geq n$.
- Given a sequence $s$ and a set $X$, we write $s \mid X$ for the subsequence of $s$ which consists of elements in $X$. In practice, we often have $s \in \mathbb{Z}^\ast$ with $\mathbb{Z} = X + Y$ for some set $Y$; in such a case, we abuse the notation: The operation deletes the “tags” for the disjoint union, so that we have $s \mid X \in X^\ast$.
- For a function $f : A \to B$ and a subset $S \subseteq A$, we define $f \mid S : S \to B$ to be the restriction of $f$ to $S$.
- For a pair of sets $A, B$, we write $B^A$ for the set of functions from $A$ to $B$.
- We write $A \cong B$ if there is an isomorphism $f : A \to B$ between objects $A$ and $B$ in a category $\mathcal{C}$.
- Let $X$ be a set, and $\sim$ an equivalence relation on $X$. For a function $f : X \to Y$, a relation $R \subseteq X \times X$, and a set $S \subseteq X^\ast$, a finite sequences of elements in $X$, we define:
  - A function $f_\sim : X/_\sim \to Y$ by $f : [x] \mapsto f(x)$ if $f(x) = f(x')$ for all $x, x' \in X$ with $x \sim x'$.
  - A relation $R_\sim \subseteq X/_\sim \times X/_\sim$ by $([x_1], [x_2]) \in R_\sim \overset{\text{df}}{=} (x_1, x_2) \in R$ if $(x_1, x_2) \in R \iff (x_1', x_2') \in R$ for all $x_1, x_1', x_2, x_2' \in X$ with $x_1 \sim x_1' \land x_2 \sim x_2'$.
  - A set $S_\sim \subseteq (X/_\sim)^\ast$ by $[x_1] \ldots [x_k] \in S_\sim \overset{\text{df}}{=} x_1 \ldots x_k \in S$ if $x_1 \ldots x_k \in S \iff x_1' \ldots x_k' \in S$ for all $k \in \mathbb{N}, x_1, \ldots, x_k, x_1', \ldots, x_k' \in X$ with $x_i \sim x_i'$ for $i = 1, \ldots, k$.  

5
2.1 Games and Strategies

Our variant of games and strategies will be based on the existing ones, specifically the ones defined in [McC98], which we call MC-games and innocent strategies. We select this variant because it is relatively less restrictive, which is important to interpret various constructions in intuitionistic type theory, e.g., if strategies are restricted to history-free ones as in [AJM00], then we cannot interpret sum types, as explained in [McC98]. Also, MC-games in some sense combine good points of the two best-known variants, AJM-games [AJM00] and HO-games [HO00].

In this section, we quickly review the relevant parts of MC-games and innocent strategies, which from now on we call games and strategies.

2.1.1 Arenas

Games are based on a preliminary notion, called arenas.

- **Definition 2.1.1 (Arenas [McC98]).** An arena is a triple $G = (M_G, \lambda_G, \vdash_G)$, where:
  
  - $M_G$ is a set, whose elements are called moves.
  
  - $\lambda_G$ is a function from $M_G$ to $\{O, P\} \times \{Q, A\}$, where $O, P, Q, A$ are some distinguished symbols, called the labeling function.
  
  - $\vdash_G$ is a subset of the set $\left(\{\star\} \times M_G\right) \times M_G$, where $\star$ is an arbitrary element, called the enabling relation, which satisfies the following conditions:
    
    - (E1) If $\star \vdash_G m$, then $\lambda_G(m) = OQ$ and $n \vdash_G m \iff n = \star$
    
    - (E2) If $m \vdash_G n$ and $\lambda_G^Q(n) = A$, then $\lambda_G^A(m) = Q$
    
    - (E3) If $m \vdash_G n$ and $m \neq \star$, then $\lambda_G^O(m) \neq \lambda_G^P(n)$

  in which we used the following notations:
  
  - $\lambda_G^O \equiv \pi_1 \circ \lambda_G : M_G \rightarrow \{O, P\}$
  
  - $\lambda_G^Q \equiv \pi_2 \circ \lambda_G : M_G \rightarrow \{Q, A\}$.

- **Convention.** A move $m \in M_G$ of an arena $G$ is called
  
  - initial if $\star \vdash_G m$
  
  - an O-move if $\lambda_G^O(m) = O$, and a P-move if $\lambda_G^P(m) = P$
  
  - a question if $\lambda_G^Q(m) = Q$, and an answer if $\lambda_G^A(m) = A$.

2.1.2 Justified Sequences

Given an arena, we are interested in certain finite sequences of the moves, called justified sequences.

- **Definition 2.1.2 (Justified sequences and justifiers [HO00, McC98]).** A justified sequence in an arena $G$ is a finite sequence $s \in M^*_G$, in which each non-initial move $m$ is associated with (or points at) a move $J_s(m)$, called the justifier of $m$ in $s$, that occurs previously in $s$ and satisfies $J_s(m) \vdash_G m$. We also say that $m$ is justified by $J_s(m)$.

  The idea is that each non-initial move in a justified sequence must be made for a previous move, called its justifier. We often omit subscripts $s$ in $J_s$. 


2.1.3 Views, Legal Positions, and Threads

We proceed to define the remaining preliminary definitions: views, legal positions, and threads.

Definition 2.1.3 (Views [HO00, McC98]). Given a justified sequence $s$ in an arena $G$, we define the Player view (or the P-view for short) $\lceil s \rceil_G$ and the Opponent view (or the O-view for short) $\lfloor s \rfloor_G$ by induction on the length of $s$ as follows:

- $\lceil \epsilon \rceil_G \overset{df}{=} \epsilon$
- $\lceil sm \rceil_G \overset{df}{=} \lceil s \rceil_G.m$, if $m$ is a P-move
- $\lceil sm \rceil_G \overset{df}{=} m$, if $m$ is initial
- $\lfloor smt \rceil_G \overset{df}{=} \lfloor s \rfloor_G.mm$, if $n$ is an O-move with $J_{smt}(n) = m$
- $\lfloor \epsilon \rfloor_G \overset{df}{=} \epsilon$
- $\lfloor sm \rfloor_G \overset{df}{=} \lfloor s \rfloor_G.m$, if $m$ is an O-move
- $\lfloor smt \rceil_G \overset{df}{=} \lfloor s \rfloor_G.mm$, if $n$ is a P-move with $J_{smt}(n) = m$.

Conceptually, views are “currently relevant” parts of previous moves; see [HO00, McC98] for the details.

In [McC98], an arena specifies the basic rules of a game in terms of certain justified sequences, called legal positions, in the sense that every “play” of the game must be a legal position (but the converse does not necessarily hold).

Definition 2.1.4 (Legal positions [McC98]). A legal position in an arena $G$ is a sequence $s \in M^*_G$ that satisfies the following conditions:

- Justification. $s$ is a justified sequence in $G$.
- Alternation. If $s = s_1mns_2$, then $\lambda^OP_G(m) \neq \lambda^OP_G(n)$.
- Bracketing. If $tguu \preceq s$, where the question $q$ is “answered” by the answer $a$ (i.e., $q$ justifies $a$), then there is no “unanswered” question in $u$.
- Visibility. If $s$ is of the form $s = tmu$ with $m$ non-initial, then:
  - if $m$ is a P-move, then the justifier $J_s(m)$ occurs in $\lceil t \rceil_G$
  - if $m$ is an O-move, then the justifier $J_s(m)$ occurs in $\lfloor t \rfloor_G$.

The set of all legal positions of an arena $G$ is denoted by $L_G$.

Let us pause here and explain the idea that “an arena specifies the basic rules of a game in terms of legal positions”:

- An arena $G$ defines the moves of a game with the O/P and Q/A parities specified, and “which move can justify which”.
- The axiom (E1) sets the convention that a play must begin with an O-move, which must be a question by the obvious reason.
- The axiom (E2) states that an answer must be made for a question.
The axiom (E3) mentions that an O-move must be justified by a P-move and vice versa.

Then in terms of legal positions, an arena specifies the basic rules of a game: In a play of the game (see Definition 2.1.6), Opponent always makes the first move, and then Player and Opponent alternatively play (alternation), in which every non-initial move must be made for a previous move (justification).

Moreover, we require that an answer must be made for the most recent unanswered question (bracketing).

Finally, visibility condition states that the justifier of each non-initial move must belong to the “relevant part” of the previous moves.

Next, we define the notion of threads. In a legal position, there may be several initial moves; the legal position consists of chains of justifiers initiated by such initial moves. These chains form threads. Formally,

Definition 2.1.5 (Hereditarily justified moves and threads [McC98]). Let G be an arena, and s ∈ LG. Assume that m is an occurrence of a move in s. The chain of justifiers from m is a maximal sequence x0x1...xkm of justifiers from m, i.e., moves x0, x1, ..., xk, m of G that satisfy

J(m) = xk, J(xk) = xk−1, ..., J(x1) = x0

where x0 is initial. In this case, we say that m is hereditarily justified by the occurrence x0 of an initial move. Moreover, the subsequence of s consisting of the chains of justifiers that end with x0 is called the thread of x0 in s.

Convention. An occurrence of an initial move is often called an initial occurrence.

Notation. We introduce a convenient notation:

We write s↾n, where s is a legal position of an arena and n is an initial occurrence in s, for the thread of n in s.

More generally, we write s↾I, where s is a legal position of an arena and I is a set of initial occurrences in s, for the subsequence of s consisting of threads of initial occurrences in I.

2.1.4 Games

We are now ready to define the notion of games.

Definition 2.1.6 (Games [McC98]). A game is a quadruple G = (MG, λG, ⊬G, PG), where:

The triple (MG, λG, ⊬G) forms an arena (also denoted by G).

PG is a subset of LG, whose elements are called the valid positions (or plays) of G, that satisfies:

(V1) PG is non-empty and prefix-closed.

(V2) If s ∈ PG and I is a set of initial occurrences in s, then s↾I ∈ PG.

The axiom (V1) talks about the natural phenomenon that each non-empty play must have the previous play, while the axiom (V2) corresponds to the idea that a play consists of several plays that are independently developed.

Note that it is possible to have a game G such that there is some move m ∈ MG that does not occur in any play, or some enabled pair m ⊬G n is not used for a justification in a play. For technical convenience, we would like to prohibit such unused structures; in other words, we shall focus on economical games:
Definition 2.1.7 (Economical games). A game $G$ is said to be economical if every move $m \in M_G$ appears in a valid position of $G$, and every enabled pair $x \vdash_G y$ occurs as a non-initial move $y$ and its justifier $x$ in a valid position of $G$.

Clearly, we will not lose any important generality with the following convention:

**Convention.** From now on, we exclusively focus on economical games, and the term “games” refers to economical games by default.

We will assume that games are always well-opened:

Definition 2.1.8 (Well-opened games [AJM00, McC98]). A game $G$ is said to be well-opened if $sm \in P_G$ with $m$ initial implies $s = \epsilon$.

That is, a well-opened game is a game in which each play has at most one initial move.

Next, we introduce the notion of subgames, a sort of a “substructure-relation” such as subgroups, subcategories, etc.

Definition 2.1.9 (Subgames). A subgame of a game $G$ is a game $H$ that satisfies:

1. $M_H \subseteq M_G$
2. $\lambda_H = \lambda_G \mid M_H$
3. $\vdash_H = \vdash_G \cap ((\{\star\} + M_H) \times M_H)$
4. $P_H = P_G \cap M_H$.

In this case, we write $H \leq G$.

We further define a stronger notion of subgames.

Definition 2.1.10 (Total subgames). A subgame $H$ of a game $G$ is said to be a total subgame of $G$ and written $H \leq_{\text{tot}} G$ if every play $s \in P_H$ that is maximal in $P_H$ (with respect to the partial order of prefix relation $\preceq$ on $P_H$) is also maximal in $P_G$.

2.1.5 Constructions on Games

Here, we quickly review the existing constructions on games. Again, our standard reference is [McC98].

We begin with the tensor product of games. Conceptually, the tensor product $A \otimes B$ is the game in which the component games $A$ and $B$ are played “in parallel without communication”.

Definition 2.1.11 (Tensor product [AJ94, McC98]). Given games $A$ and $B$, we define their tensor product $A \otimes B$ as follows:

1. $M_{A \otimes B} \equiv M_A + M_B$
2. $\lambda_{A \otimes B} \equiv [\lambda_A, \lambda_B]$
3. $\vdash_{A \otimes B} \equiv \vdash_A \cup \vdash_B$
4. $P_{A \otimes B} \equiv \{s \in L_{A \otimes B} \mid s \upharpoonright A \in P_A, s \upharpoonright B \in P_B\}$.

Of course, all constructions on games in this paper preserve this property.
Next, we consider linear implication.

**Definition 2.1.12** (Linear implication [AJ94, McC98]). Given games $A$ and $B$, we define their linear implication $A \to B$ as follows:

- $M_{A \to B} \triangleq M_A + M_B$
- $\lambda_{A \to B} \triangleq [\lambda_A, \lambda_B]$
- $\star \vdash_{A \to B} m \triangleq \star \vdash_B m$
- $m \vdash_{A \to B} n (m \neq \star) \triangleq (m \vdash_A n) \lor (m \vdash_B n) \lor (\star \vdash_B m \land \star \vdash_A n)$
- $P_{A \to B} \triangleq \{ s \in L_{A \to B} \mid s \upharpoonright A \in P_A, s \upharpoonright B = \epsilon \} \cup \{ s \in L_{A \& B} \mid s \upharpoonright A = \epsilon, s \upharpoonright B \in P_B \}.$

where $\lambda_G \triangleq (\lambda_{G}^{OP}, \lambda_{G}^{OA})$ and $\lambda_{G}^{OP}(m) \triangleq \begin{cases} P & \text{if } \lambda_{G}^{OP}(m) = O \\ 0 & \text{otherwise} \end{cases}$ for any game $G.$

**Convention.** For any game $G$, we shall not distinguish the linear implication $I \to G$ and $G$, where $I = (\emptyset, \emptyset, \emptyset, \{ \epsilon \})$ is the empty game.

The construction of products is the categorical product in the cartesian closed category of MC-games and innocent strategies (see [McC98]).

**Definition 2.1.13** (Product [HO00, McC98]). Given games $A$ and $B$, we define their product $A \& B$ as follows:

- $M_{A \& B} \triangleq M_A + M_B$
- $\lambda_{A \& B} \triangleq [\lambda_A, \lambda_B]$
- $\vdash_{A \& B} \triangleq \vdash_A \cup \vdash_B$
- $P_{A \& B} \triangleq \{ s \in L_{A \& B} \mid s \upharpoonright A \in P_A, s \upharpoonright B = \epsilon \} \cup \{ s \in L_{A \& B} \mid s \upharpoonright A = \epsilon, s \upharpoonright B \in P_B \}.$

Like AJM- and MC-games [AM00, McC98], the construction of exponential will be crucial when we equip our category of games and strategies with a cartesian closed structure.

**Definition 2.1.14** (Exponential [HO00, McC98]). For any game $A$, we define its exponential $!A$ as follows:

- $M_{!A} \triangleq M_A$
- $\lambda_{!A} \triangleq \lambda_A$
- $\vdash_{!A} \triangleq \vdash_A$
- $P_{!A} \triangleq \{ s \in L_{!A} \mid s \upharpoonright m \in P_A \text{ for each initial occurrence } m \text{ in } s \}.$

**Notation.** Notationally, exponential precedes any other operation; also, tensor product and product both precede linear implication. E.g., $!A \to B$ means $(!A) \to B$, and $A \otimes!B \to !C \& D$ means $(A \otimes (!B)) \to ((!C)\&D)$, etc.
2.1.6 Strategies

We proceed to define strategies. Roughly, a strategy is what tells Player which move she should make next.

Definition 2.1.15 (Strategies \[AJ94, AJM00, HO00, McC98\]). A strategy \(\sigma\) on a game \(G\) is a set of even-length valid positions of \(G\), satisfying the following conditions:

- (S1) It is non-empty and “even-prefix-closed”: \(s_{mn} \in \sigma \Rightarrow s \in \sigma\).
- (S2) It is deterministic: \(s_{mn}, s_{mn}' \in \sigma \Rightarrow n = n' \land J_{s_{mn}}(n) = J_{s_{mn}'}(n')\).

We write \(\sigma : G\) to indicate that \(\sigma\) is a well-defined strategy on the game \(G\).

Convention. From now on, we shall abbreviate the condition \(n = n' \land J_{s_{mn}}(n) = J_{s_{mn}'}(n')\) as \(n = n'\) (i.e., the equality of justifiers will be implicit).

In general, a strategy can be “partial” in the sense that it may not have any response at some odd-length valid position. Thus, it makes sense to define:

Definition 2.1.16 (Total strategies \[A^+97\]). A strategy \(\sigma : G\) is called total if it satisfies:

\[(\text{tot}) \ s \in \sigma \land s_{m} \in P_{G} \Rightarrow \exists n \in M_{G}. s_{mn} \in \sigma.\]

Because we are interested in total type theory, i.e., a type theory in which every computation terminates, we need to focus on total strategies to establish a definability result. However, we are aiming to obtain a soundness (not necessarily completeness) result in the present paper, we shall address this problem as a future work, and we will not consider total strategies in depth in this paper.

A strategy \(\sigma : G\), where \(G\) is well-opened, can be seen as a particular subgame of \(G\). To establish this fact rigorously, we need the following definition:

Definition 2.1.17 (Strategies as trees). Let \(\sigma\) be a strategy on a game \(G\). We define:

\[\sigma_{G} \overset{\text{df}}{=} \sigma \cup \{s_{m} \in P_{G} \mid s \in \sigma, m \in M_{G}\}\]

and call it the tree-form of \(\sigma\) with respect to \(G\).

We often omit the subscript \(G\) and just write \(\sigma\) when the underlying game \(G\) is obvious. Clearly, we may recover \(\sigma\) from \(\sigma_{G}\) by removing all the odd-length plays. Thus, \(\sigma\) and \(\sigma_{G}\) are essentially the same (in the context of \(G\)), just in different forms.

Note in particular that \(\sigma\) is a non-empty, prefix-closed subset of \(P_{G}\). We may in fact characterize strategies on a game \(G\) as follows:

Lemma 2.1.18 (Strategies in second-form). For any game \(G\), there is a one-to-one correspondence between strategies \(\sigma : G\) and subsets \(S \subseteq P_{G}\) that is:

- (tree) Non-empty and prefix-closed (i.e., \(s_{m} \in S \Rightarrow s \in S\)).
- (edet) Deterministic on even-length positions (i.e., \(s_{mn}, s_{mn}' \in S^{\text{even}} \Rightarrow n = n'\)).
- (oinc) Inclusive on odd-length positions (i.e., \(s \in S^{\text{even}} \land s_{m} \in P_{G} \Rightarrow s_{mn} \in S\)).
Proof. First, it is straightforward to see that, for each strategy \( \sigma : G \), the subset \( \sigma \subseteq P_G \) satisfies the three conditions of the lemma; e.g., \( \sigma \) is non-empty because \( \epsilon \in \sigma \), and it is prefix-closed: For any \( sm \in \sigma \), if \( s \in \sigma \), then \( s \in \sigma \), otherwise, i.e., \( sm \in \sigma \), we may write \( s = tn \in P_G \) for some \( t \in \sigma \), \( n \in M_G \), whence \( s \in \sigma \).

For the converse, assume that a subset \( S \subseteq P_G \) satisfies the three conditions. We then have \( S^\text{even} \subseteq P_G^\text{even} \).

- \( S^\text{even} \) is non-empty as \( \epsilon \in S^\text{even} \), and it is even-prefix-closed: If \( smn \in S^\text{even} \), then \( smn \in S \), whence \( s \in S \) and so \( s \in S^\text{even} \) because \( S \) is prefix-closed.

- \( S^\text{even} \) is deterministic: If \( smn, smn' \in S^\text{even} \), then \( n = n' \) by the determinacy on even-length positions of \( S \).

Finally, we show that these constructions are mutually inverses. Clearly \( (\sigma)^\text{even} = \sigma \) for all \( \sigma : G \). It remains to establish the equation \( S^\text{even} = S \) for all \( S \subseteq P_G \) satisfying the three conditions. Let \( S \subseteq P_G \) be such a subset. It is immediate that \( s \in S^\text{even} \iff s \in S \) for any even-length play \( s \in P_G^\text{even} \). If \( tm \in S^\text{even} \) is of odd-length, then \( t \in S^\text{even} \) and \( tm \in P_G \), so we have \( tm \in S \) as \( S \) is inclusive on odd-length positions. Conversely, if \( un \in S \) is of odd-length, then \( u \in S^\text{even} \) and \( un \in P_G \), whence \( un \in S^\text{even} \).

Thanks to the lemma, we employ:

- Convention. From now on, we identify strategies on a game \( G \) with subsets of \( P_G \) that are non-empty, prefix-closed, deterministic on even-length positions, and inclusive on odd-length positions.

Now we establish:

\begin{itemize}
  \item Lemma 2.1.19 (Strategies as subgames). For any (resp. total) strategy \( \sigma \) on a well-opened game \( G \), the structure \( \langle M_\sigma, \lambda_\sigma, \vdash_\sigma, \emptyset \rangle \) is a (resp. total) subgame of \( G \), where \( M_\sigma \subseteq M_G \) is the set of moves of \( G \) that appear in \( \sigma \), \( \lambda_\sigma \overset{\text{df}}{=} \lambda_G \upharpoonright M_\sigma \) and \( \vdash_\sigma \overset{\text{df}}{=} \vdash_G \cap (\{\star\} \times M_\sigma) \times M_\sigma \).
\end{itemize}

Proof. We only show that \( \sigma \) satisfies the axioms (V1) and (V2) because it is easy to verify the conditions on the other components. We have shown (V1) in the proof of Lemma 2.1.18. And (V2) is trivially satisfied because \( G \) is well-opened. Finally, it is easy to check the additional condition of total strategies.

- Convention. In the rest of the paper, strategies \( \sigma : G \) are often identified with the subgames \( \langle M_\sigma, \lambda_\sigma, \vdash_\sigma, \emptyset \rangle \subseteq G \) established above, and abusing the notation we write \( \sigma \) for such subgames. In this sense, the relation \( \sigma : G \) is a particular kind of the subgame relation \( \sigma \subseteq G \).

### 2.1.7 Constructions on Strategies

Next, we review the existing constructions on strategies in [McC98] that are rather standard in the literature of game semantics.

One of the most basic strategies is the so-called *copy-cat strategies*, which basically “copy and paste” the last O-moves.

---

7Strictly speaking, \( \vdash_\sigma \) is defined to be a subset of \( \vdash_G \cap ([\{\star\}] + M_\sigma) \times M_\sigma \) in such a way that the structure \( \langle M_\sigma, \lambda_\sigma, \vdash_\sigma, \emptyset \rangle \) forms an economical game.
Definition 2.1.20 (Copy-cat strategies [AJ94, AJM00, HO00, McC98]). The copy-cat strategy \( cp_A : A_1 \rightarrow A_2 \) on a game \( A \) is defined by:

\[
cp_A \overset{df}{=} \{ s \in P^\text{even}_{A_1 \rightarrow A_2} \mid \forall t \preceq s. \text{even}(t) \Rightarrow t \parallel A_1 = t \parallel A_2 \}
\]

where the subscripts 1, 2 on \( A \) are to distinguish the two copies of \( A \).

Next, to formulate the composition of strategies, it is convenient to first define the following intermediate concept:

Definition 2.1.21 (Parallel composition [A+97]). Given strategies \( \sigma : A \rightarrow B, \tau : B \rightarrow C \), we define their parallel composition \( \sigma \parallel \tau \) by:

\[
\sigma \parallel \tau \overset{df}{=} \{ s \in M^* \mid s \parallel A, B_1 \in \sigma, s \parallel B_2, C \in \tau, s \parallel B_1, B_2 \in \text{pr}_B \}
\]

where \( \text{pr}_B \overset{df}{=} \{ s \in P_{B_1 \rightarrow B_2} \mid \forall t \preceq s. \text{even}(t) \Rightarrow t \parallel B_1 = t \parallel B_2 \} \), \( \sigma \overset{df}{=} M_{A} + M_{B_1} + M_{B_2} + M_{C} \), and \( B_1, B_2 \) are just two copies of \( B \).

Remark. Parallel composition is only a preliminary definition for the official composition of the category of games and strategies in [AJM00, McC98]; it is not a well-defined operation (however, for “dynamic” games and strategies in [YA16], it is well-defined and rather plays an important role).

We now define the composition of strategies, which can be phrased as “parallel composition plus hiding” [A+97].

Definition 2.1.22 (Composition [AJ94, A+97, McC98]). Given strategies \( \sigma : A \rightarrow B, \tau : B \rightarrow C \), we define their composition \( \sigma \tau : A \rightarrow C \) by \( \sigma \tau \overset{df}{=} \{ s \parallel A, C \parallel \sigma \parallel \tau \} \).

Notation. We also write \( \tau \circ \sigma \) for a composition \( \sigma \tau \).

Remark. Strictly speaking, the above definition is incomplete as it does not specify the resulting justifiers. For such a detail, see [McC98].

Next, we define tensor product of strategies, which is the “disjoint union of strategies without interaction”.

Definition 2.1.23 (Tensor product [AJ94, AJM00, McC98]). Given strategies \( \sigma : A \rightarrow C, \tau : B \rightarrow D \), their tensor product \( \sigma \otimes \tau : A \otimes B \rightarrow C \otimes D \) is defined by:

\[
\sigma \otimes \tau \overset{df}{=} \{ s \in L \mid s \parallel A, C \in \sigma, s \parallel B, D \in \tau \}
\]

where \( L \overset{df}{=} L_{A \otimes B \rightarrow C \otimes D} \).

We proceed to define the construction of paring.

Definition 2.1.24 (Paring [AJM00, McC98]). Given strategies \( \sigma : C \rightarrow A, \tau : C \rightarrow B \), we define their paring \( \langle \sigma, \tau \rangle : C \rightarrow A \& B \) by:

\[
\langle \sigma, \tau \rangle \overset{df}{=} \{ s \in L \mid s \parallel C, A \in \sigma, s \parallel B = \epsilon \} \cup \{ s \in L \mid s \parallel C, B \in \tau, s \parallel A = \epsilon \}
\]

where \( L \overset{df}{=} L_{C \rightarrow A \& B} \).

Next, we define a construction which is fundamental when we equip the categories of games and strategies in [AJM00, McC98] with a cartesian closed structure:
Definition 2.1.25 (Promotion [AJM00, McC98]). Given a strategy $\sigma : !A \rightarrow B$, we define its promotion $\sigma^! : !A \rightarrow !B$ by $\sigma^! \overset{df}{=} \{ s \in L_{!A \rightarrow !B} \mid s \upharpoonright m \in \sigma \text{ for all initial occurrences } m \text{ in } s \}$. 

Intuitively, the promotion $\sigma^!$ is the strategy which plays as $\sigma$ for each thread.

Remark. Before defining derelictions, we make a brief detour:

- If the exponential $!$ were a comonad (in the form of a co-Kleisli triple), then there would be a strategy $\text{der}_A : !A \rightarrow A$ which should be called the dereliction on $A$, satisfying $\text{der}_A^! = c_{p_A}$ and $\sigma^! ; \text{der}_B = \sigma$, for any games $A, B$ and strategy $\sigma : !A \rightarrow B$.

- It appears that we may take the copy-cat strategy $c_{p_A}$ as $\text{der}_A$; however, it does not work for an arbitrary game $A$, as described in [McC98]. In fact, we have to require games to be well-opened.

- Note that if $B$ is well-opened, then so is the linear implication $A \rightarrow B$ for any game $A$.

- In the cartesian closed category of games and strategies in [McC98], all games are well-opened, and exponentials are given by $A \rightarrow B \overset{df}{=} !A \rightarrow !B$, which are thus well-opened.

- Also, note that even if a game $A$ is well-opened, its exponential $!A$ is not. However, in the cartesian closed category, exponential $!$ is not an allowed construction; thus the objects remain well-opened.

Now we are ready to define derelictions:

Definition 2.1.26 (Derelictions [AJM00, McC98]). Let $A$ be a well-opened game. Then we define a strategy $\text{der}_A : !A \rightarrow A$, called the dereliction on $A$, to be the copy-cat strategy $c_{p_A}$. 

14
3 Predicative Games

One of the main problems in interpreting intuitionistic type theory in terms of games and strategies is how to interpret universes as games. e.g., the game-theoretic model in [AV15] does not interpret universes. Our solution is to allow “names” of games to be moves of other games.

However, a naive formulation would result in a Russell-like paradox: We would have a “game of all games”. To circumvent this problem, we type moves and games, and propose a kind of a “standard form” of games. The resulting games and strategies, called predicative games and generalized strategies, have enough structures to interpret the type theory.

Our variant of games and strategies makes several generalizations of the existing variants. It generalizes the relation “σ is a strategy on a game G” in such a way that it makes sense to say that “a predicative game A is a generalized strategy on another predicative game B”, which interprets the phenomenon in the type theory that every type is a term of a universe. Additionally, it generalizes the existing games in such a way that a predicative game can be a sort of a “family of games”, which is to interpret constructions such as dependent function space and dependent pair space in the type theory. Accordingly, all the constructions on games and strategies in Sections 2.1.5, 2.1.7 are generalized and unified in a systematic manner, in which a new inductive structure arises.

This section presents predicative games and generalized strategies; in particular, we define universe games that interpret (the hierarchy of) universes as a certain kind of predicative games.

3.1 Typed Games

As mentioned above, our games may have “names” of games as moves. Thus, a move is either a “mere” move or the “name” of another game. Importantly, such a distinction should not be ambiguous because, conceptually, each move in a game must have a definite role. This naturally leads us to require that moves are “typed”, which also works to “type” games in order to avoid a Russell-like paradox as we shall see shortly.

We begin with typing moves.

Definition 3.1.1 (Typed moves). For a game, a move is said to be typed if it is a pair $(m, k)$ of some object $m$ and a natural number $k \in \mathbb{N}$, which is usually written $[m]_k$. A typed move $[m]_k$ is more specifically called a $k$-type move, and $k$ is said to be the type of the move. In particular, a $0$-type move is called a mere move.

Our intension is as follows: A mere move is just a move of a game in the usual sense, and a $(k+1)$-type move is the “name” of another game (whose moves are all typed) such that the supremum of the types of the moves is $k$ (see Definition 3.1.2 below).

Based on types of moves, we now define the notion of typed games.

Definition 3.1.2 (Typed games). A typed game is a game whose moves are all typed. The type of a typed game $G$, written $\mathcal{T}(G)$, is defined by:

\[
\mathcal{T}(G) \overset{\text{df}}{=} \begin{cases} 
1 & \text{if } M_G = \emptyset \\
\sup\{k+1 | [m]_k \in M_G\} & \text{otherwise.}
\end{cases}
\]

More specifically, $G$ is called a $\mathcal{T}(G)$-type game.

Because a universe $\mathcal{U}$ is a type, it should be interpreted as a game $\llbracket \mathcal{U} \rrbracket$. But then, the game $\llbracket A \rrbracket$ that interpretes a (small) type $A : \mathcal{U}$ must be somehow seen as a strategy on $\llbracket \mathcal{U} \rrbracket$. . .
One may wonder if the type of a typed game can be transfinite; however, as we shall see, the type of a predicative game is always a natural number. Also, we have defined $T(G) \overset{df}{=} 1$ if $M_G = \emptyset$ as such a game $G$ can be seen as a trivial case of a game that has mere moves only.

We now define “names” of games:

- **Definition 3.1.3** (Names of games). The name of a typed game $G$, written $\mathcal{N}(G)$, is the pair $(G, T(G))$ of $G$ (as a set) itself and its type $T(G)$.

  Note that the name of a typed game can be a typed move; however, that name cannot be a move of the game itself because of the typing method, which is how we shall avoid a Russell-like paradox (see Proposition 3.3.3 below).

### 3.2 1-predicative and 2-predicative Games

In the next section, we shall define a certain kind of typed games, called predicative games, which are the games to interpret types of intuitionistic type theory. For this, we define in this section a preliminary version of predicative games with the lowest and second lowest types, called 1-predicative games and 2-predicative games, respectively, as a preparation. As we shall see, they are essentially strategies and games in the usual game semantics.

- **Remark.** This section serves as a “bridge” between the usual notion of games and strategies on one hand and predicative games and generalized strategies in the next section on the other hand. Thus, the reader may safely skip it and jump to the next section.

First, we define essential equality on moves, legal positions, and games:

- **Definition 3.2.1** (Essential equality on moves, legal positions, and games). For a game $G$, moves $m, m' \in M_G$ are said to be essentially equal and written $m \equiv m'$ if they are the same elements up to the “tags” for disjoint union. Also, legal positions $s, s' \in L_G$, where $s = m_1 \ldots m_k, s' = m'_1 \ldots m'_k$, are said to be essentially equal and written $s \equiv s'$ if $m_i \equiv m'_i$ for $i = 1, \ldots, k$ and their justifiers are corresponding. Moreover, games $G, G'$ are said to be essentially equal and written $G \equiv G'$ if they are the same games when we identify essentially equal moves.

When considering essential equality between games $G$, we must ensure that the structure $G/\equiv = (M_G/\equiv, \lambda_G/\equiv, \vdash_G/\equiv, P_G/\equiv)$ obtained from $G$ by identifying essentially equal moves is a well-defined game, i.e., $m \equiv m'$ implies $\lambda_G(m) = \lambda_G(m'), m \equiv m' \wedge n \equiv n'$ implies $\vdash_G m \iff \vdash_G m'$ and $m \vdash_G n \iff m' \vdash_G n'$, and $s \equiv s'$ implies $s \in P_G \iff s' \in P_G$.

We then use the following definition to ensure that the resulting game satisfies these conditions when a certain construction is applied.

- **Definition 3.2.2** (Consistent sets of games). A set $S$ of games is said to be consistent if, for any $A, B \in S$, it satisfies the following:

  - $\lambda_A(m) = \lambda_B(m)$ for all $m \in M_A \cap M_B$.
  - $\star \vdash_A m \iff \star \vdash_B m$ and $m \vdash_A n \iff m \vdash_B n$ for all $m, n \in M_A \cap M_B$.
  - $s \in P_A \iff s \in P_B$ for all $s \in L_A \cap L_B$.

Now, observe the following:

- **Proposition 3.2.3** (Games as collections of strategies). A game $G$ corresponds to the set of its strategies $\sigma : G$ in the following sense:

  More precisely, $m_j$ is a justifier of $m_i$ iff $m'_j$ is a justifier of $m'_i$.  

16
1. For any game $G$, we have:
\[
G = (\sum_{\sigma \in G} \tau) / \equiv \equiv (\sum_{\sigma \in G} M) / \equiv, (\bigcup_{\tau \in G} \lambda \tau) / \equiv, (\bigcup_{\tau \in G} \lambda \tau) / \equiv.
\]
where $\lambda \tau$, $\lambda \tau$, $\lambda \tau$, $\lambda \tau$, $\lambda \tau$, $\lambda \tau$, $\lambda \tau$, $\lambda \tau$, $\lambda \tau$ are the obvious modifications of $\lambda \tau$, $\lambda \tau$, $\lambda \tau$, $\lambda \tau$, $\lambda \tau$, $\lambda \tau$, $\lambda \tau$, $\lambda \tau$, $\lambda \tau$ up to the “tags” for the disjoint union $\sum_{\sigma \in G} M$ of moves, and the quotient $/ \equiv$ is taken only for the essential equality that arises when taking the disjoint union $\sum_{\sigma \in G} M$.

2. For any consistent set $S$ of strategies (not necessarily on the same game), the game
\[
\{S / \equiv\} \equiv (\sum_{\sigma \in G} \tau) / \equiv, (\bigcup_{\tau \in G} \lambda \tau) / \equiv, (\bigcup_{\tau \in G} \lambda \tau) / \equiv, (\bigcup_{\tau \in G} \lambda \tau) / \equiv.
\]
where the symbol $\{S / \equiv\}$ has the same meaning as in the clause 1 and the quotient $/ \equiv$ is taken only for the essential equality that arises when taking the disjoint union $\sum_{\tau \in G} M$, is well-defined and has $\{S / \equiv\}$ as its strategy for all $\tau \in S$. Moreover, the converse inclusion also holds (i.e., every strategy on $\{S / \equiv\}$ is in $S$) if $S = \{\sigma \mid \sigma : G\}$ for some game $G$.

Proof. Let $G$ be a game. We first show the clause 1. Note that $\equiv$ is clearly an equivalence relation on $\sum_{\sigma \in G} M$, and the quotient of each component is well-defined. For the sets of moves and the enabling relations, the inclusion $\supseteq$ is obvious, but the converse inclusion holds as well because $G$ is economical and by the equation $P_G = \bigcup_{\sigma \in G} \sigma$ established below. Then, the labelling functions clearly coincide.

For the valid positions, it suffices to show the equation $P_G = \bigcup_{\sigma \in G} \sigma$. One direction $\bigcup_{\sigma \in G} \sigma \subseteq P_G$ is immediate because $\sigma : G$ implies $\sigma \subseteq P_G$ (see Lemma 2.1.18). For the other inclusion, assume that $s \in P_G$; we shall show $s \in \sigma$ for some $\sigma : G$. If $s$ is of even-length, then we may just take $\sigma \equiv \text{pref}((s))$. If $s$ is of odd-length, say $s = t m$ with $\text{even}(t)$, then we take $\sigma \equiv \text{pref}((t))$. It remains to verify the clause 2. Again, $\equiv$ is an equivalence relation on $\sum_{\tau \in G} M$, and the quotient of each component is well-defined because the set $S$ is consistent. Then the first half of the clause 2 is obvious; and the second half immediately follows from the clause 1.\]

Remark. Of course, when we consider strategies, we usually have the underlying game in our minds. What we are saying here is that we may form a game from strategies even without such an a priori concept of underlying games. Usually, the notion of strategies in game semantics comes after that of games, but the proposition enables us to reverse the order.\[\]

\begin{itemize}
\item Notation. For a finite sequence of moves $s = m_1 m_2 \ldots m_n$ and an integer $k \in \mathbb{Z}$, we define $s_k \equiv [m_1] k, [m_2] k \ldots [m_n] k$. This applies for infinite sequences as well in the obvious way.
\end{itemize}
Now, we define 1-predicative games, which are essentially strategies on well-opened games.

**Definition 3.2.4** (1-predicative games (preliminary)). A 1-predicative game is a game \( \sigma_1 \), where \( \sigma : G \) is any strategy on a well-opened game \( G \), defined as follows:

- \( M_{\sigma_1} \overset{\text{df}}{=} \{ [m]_0 \mid m \in M_\sigma \} \)
- \( \lambda_{\sigma_1} : [m]_0 \mapsto \lambda_{\sigma}(m) \)
- \( \vdash_{\sigma_1} \overset{\text{df}}{=} \{ (\star, [m]_0) \mid \star \vdash_{\sigma} m \} \cup \{ ([m]_0, [n]_0) \mid m \vdash_{\sigma} n \} \)
- \( P_{\sigma_1} \overset{\text{df}}{=} \{ s_0 \mid s \in \sigma \} \), where the justifiers are preserved (i.e., \( J_{s_0}([m]_0) = J_s(m)_0 \) for each non-initial move \( [m]_0 \) in \( s_0 \)).

**Remark.** We put the term "preliminary" here because we shall present a different (but equivalent) definition of 1-predicative games in the next section, which we shall take as "official".

**Lemma 3.2.5** (1-predicative lemma). A 1-predicative game is a well-defined 1-type game that is well-opened.

**Proof.** Clear from the definition.

Next, we define 2-predicative games, which are essentially well-opened games in [McC98].

**Definition 3.2.6** (2-predicative games (preliminary)). A 2-predicative game is a game \( St(G) \), where \( G \) is a well-opened game, defined as follows:

- \( M_{St(G)} \overset{\text{df}}{=} \{ q_G \} + \sum_{G,G} (\{ N(\sigma_1) \} \cup M_{\sigma_1}) \), where \( q_G \overset{\text{df}}{=} [0]_0 \).
- \( \lambda_{St(G)} : q_G \mapsto OQ, N(\sigma_1) \mapsto PA, ([m]_0 \in M_{\sigma_1}) \mapsto \lambda_{\sigma}(m) \).
- The enabling relation is defined by:

\[
\vdash_{St(G)} \overset{\text{df}}{=} \{ (\star, q_G) \} \cup \{ (q_G, N(\sigma_1)) \mid \sigma : G \} \cup \{ ([m]_0, [n]_0) \mid m \vdash_{\sigma} n \} \\
\cup \bigcup_{\sigma : G} (\vdash_{\sigma_1} \cap (M_{\sigma_1} \times M_{\sigma_1})).
\]

- \( P_{St(G)} \overset{\text{df}}{=} \text{pref}([q_G, N(\sigma_1), s_0 \mid \sigma, s \in \sigma]), \) where \( N(\sigma_1) \) is justified by \( q_G \), and moves \( [m]_0 \) in \( s_0 \) such that \( m \) is initial in \( G \) are justified by \( N(\sigma_1) \).

**Remark.** Again, we have the term "preliminary" because we will give a more generalized definition of 2-predicative games in the next section, which we shall take as "official".

In view of Proposition 3.2.3 (the clause 1), a 2-predicative game \( St(G) \) is almost essentially equal to the well-opened game \( G \); also by Proposition 3.2.3 (the clause 2), strategies on \( St(G) \) and strategies on \( G \) are roughly the same. The difference lies only in that \( St(G) \) is typed and starts a play with a question and answer about Player’s strategy, and a play to follow must conform the declared strategy.

As one expects, we have:

**Lemma 3.2.7** (2-predicative lemma). A 2-predicative game is a well-defined 2-type game that is well-opened.
Proof. Let $St(G)$ be a 2-predicative game. First, it is straightforward to see that $St(G)$ is a well-opened game, in which the visibility condition is satisfied because $G$ is well-opened. And it is of 2-type by the moves $N(\sigma_1)$.

To sum up, we have shown in this section that (the preliminary version of) 2-predicative and 1-predicative games are essentially well-opened games and strategies on them. With this fact, predicative games and generalized strategies, which will be defined in the next section, can be seen as a generalization of games and strategies, respectively.

### 3.3 Predicative Games and Generalized Strategies

This section presents our variant of games and strategies. We define $k$-predicative games for each $k \in \mathbb{Z}_{\geq 1}$ and generalized strategies in such a way that it makes sense to say that a $k$-predicative game is a generalized strategy on an $l$-predicative game with $k < l$. This structure will interpret the phenomenon in the type theory that a type is a term of a universe.

- **Notation.** From now on, we just write $m$ for typed moves $[m]_k$ when it is not necessary to exhibit the type $k$.

- **Definition 3.3.1** (Predicative games and generalized strategies). A 1-type game $G$ is said to be 1-predicative if it is well-opened and deterministic on even-length positions, and satisfies $M_G \subseteq \mathbb{N} \times \{0\}$. A generalized strategy (or strategy for short) on an $l$-type game $G$, $l \in \mathbb{Z}_{\geq 2}$, is a $k$-predicative game $\sigma$ with $k < l$ whose name $N(\sigma)$ occurs in $P_G$ in such a way that there is no preceding name of a typed game, i.e., for all $s. N(\sigma). t \in P_G$, $s$ contains mere moves only. In this case, we write $\sigma : G$. For each $k \in \mathbb{Z}_{\geq 2}$, a $k$-type game $G$ is said to be $k$-predicative, if it satisfies the following:

- $M_G = \{q_G\} + \sum_{\sigma.G}\{N(\sigma)\} \cup M_\sigma$, where $q_G \triangleq [0]_0$.
- $\lambda_G : q_G \mapsto OQ, N(\sigma) \mapsto PA,(m \in M_\sigma) \mapsto \lambda_\sigma(m)$.
- $\vdash_G = \{(*, q_G)\} \cup \{q_G, N(\sigma) | \sigma : G\} \cup \{(N(\sigma), m)|\sigma : G, * \vdash_\sigma m\} \cup \bigcup_{\sigma.G}(\vdash_\sigma \cap (M_\sigma \times M_\sigma))$.

- $P_G = \operatorname{pref}(\{q_G, N(\sigma).s | \sigma : G, s \in \sigma\})$, where $N(\sigma)$ is justified by $q_G$ and the initial moves of $\sigma$ in $s$ are justified by $N(\sigma)$.

A predicative game is a $k$-predicative game for some $k \in \mathbb{Z}_{\geq 2}$. Moreover, a generalized strategy with type 1 is particularly called a strict strategy.

- **Remark.** Note that predicative games are inductively defined along with their types, and moves of $1$-predicative games are required to be natural numbers (with type 0). As a consequence, the class of moves of a predicative game will never be a proper class.

In a predicative game $G$, a play starts with a question $q_G$ and answer $N(\sigma)$ about a strategy $\sigma$ to play, then Opponent and Player iterate such a communication, decreasing the type of predicative games, until they have reached at a strict strategy, and finally a play by the strict strategy follows. In other words, predicative games are a generalization of MC-games in which we enforce such a “protocol” of plays that gradually narrows down the range of possible plays.

---

1In particular, this “official” notion of 1-predicative and 2-predicative games generalizes the preliminary one in the last section.

2This restriction does not prohibit us from having basic games such as the natural numbers game (which will be introduced later) to interpret the type theory.
By Lemma 2.1.18 and Proposition 3.2.3, a 2-predicative game and 1-predicative games on \( i \) in the sense of Definitions 3.2.6, 3.2.4 can be seen as a particular kind of a 2-predicative game and generalized strategies on it in the sense of Definition 3.3.1. Moreover, a 2-predicative game in the latter sense is a generalization of a 2-predicative game in the former sense (and the usual notion of a game by Proposition 3.2.3) because its generalized strategies may not have it as a common underlying game, or they may not range over all the strategies (in the usual sense) on it. In particular, we may form a predicative game that can be seen as a “family of games”; see, e.g., Definition 5.1.2. From now on, we shall take Definition 3.3.1 as the “official” definition.

- Notation. For each \( k \in \mathbb{Z}_{\geq 2} \), we denote \( PG_k \) for the set of all \( k \)-predicative games. Also, we write \( PG_{\leq k} \) for the set of all \( i \)-predicative games with \( 2 \leq i \leq k \). Moreover, for any predicative game \( G \) and integer \( k \in \mathbb{Z}_{\geq 1} \), we write \( gs(G) \) and \( gs_k(G) \) for the set of all generalized strategies on \( G \) and its subset consisting of \( k \)-type ones only, respectively. Furthermore, \( GS_k \) (resp. \( GS_{\leq k} \)) denotes the set of all generalized strategies with \( k \)-type (resp. \( l \)-type with \( 1 \leq l \leq k \)). Additionally, we informally write \( G \in PG \) and \( S \in GS \) to mean that \( G \) is a predicative game and \( S \) is a generalized strategy, respectively.

- Remark. We have \( PG = GS_{\geq 2} \overset{df}{=} \bigcup_{k \geq 2} GS_k \) by Lemma 3.3.7 below, which corresponds to the phenomenon in the type theory that all types can be regarded as terms (of universes), but “strict” terms cannot be types.

- Convention. Note that we no longer need the terminology “1-predicative games”; it was just for the concise way of defining generalized strategies. In fact, since a 1-predicative game is not a predicative game, the term is now very confusing. Hence, from now on, we always call 1-predicative games (generalized or more precisely strict) strategies.

Note that every generalized strategy is “total” in the sense that it always has a response; provided that all the strict strategies are total. However, we actually do not lose the generality to have non-total strategies:

- **Definition 3.3.2 (Total generalized strategies).** A generalized strategy \( \sigma : G \) is said to be total if every play \( s \in \sigma \) that is maximal (with respect to the prefix relation \( \preceq \)) in \( \sigma \) is maximal in any generalized strategy \( \tau : G \) such that \( s \in \tau \) up to the “tags” for disjoint union.

It is straightforward to see that this definition coincides with the notion of “total strategies” in the usual game semantics (see Definition 2.1.16). Note that we are not concerned that much with totality of generalized strategies in the present paper because we are aiming to obtain just a soundness, not necessarily (full) completeness, result. Accordingly, we will not study totality of strategies in depth in the present paper.

As mentioned earlier, a motivation for the definition of predicative games is to obtain a game of games without a Russell-like paradox. We now establish the fact that they are in fact paradox-free.

- **Proposition 3.3.3 (Paradox-free).** The name of a predicative game is not a move of the game, i.e., if \( G \) is a predicative game, then \( N(G) \notin MG \).

**Proof.** Let \( G \) be a \( k \)-predicative game with \( k \in \mathbb{Z}_{\geq 2} \). By the definition, every move \([m]_l \in MG \) satisfies \( l < k \). Thus, since the name \( N(G) \) has type \( k \), it cannot be a move in \( MG \). 

This result is certainly assuring: If we did not type moves nor games, and defined the name of each predicative game \( G \) to be, e.g., \( G \) itself, then we could define a (type-free) predicative

---

13 Strictly speaking, we need to ignore the first pair of a question and answer.

14 In fact, it is too general to establish a definability result, but it works to just obtain a (not necessarily fully complete) model, which is the aim of the present paper.
game \( P \) by \( \text{gs}(P) \overset{df}{=} \{ G \in PG \mid G \notin \text{gs}(G) \} \). This can be seen as a game-theoretic variant of the famous Russell’s paradox: If \( P \in \text{gs}(P) \), then \( P \notin \text{gs}(P) \); and if \( P \notin \text{gs}(P) \), then \( P \in \text{gs}(P) \).

Next, we introduce two very convenient constructions.

**Definition 3.3.4** (Parallel union of predicative games). Given an integer \( k \in \mathbb{Z}_{\geq 2} \) and a set \( S \subseteq PG_{\leq k} \) of \( i \)-predicative games with \( 2 \leq i \leq k \), we define the predicative game \( S \), called the **parallel union** of \( S \), as follows:

- \( M_{S} = \{ q_{S} \} + \sum_{G \in S}(M_{G} \setminus \{ q_{G} \}) \), where \( q_{S} = [0]_{0} \).
- \( \lambda_{S} : q_{S} \mapsto G, (m \in M_{G} \setminus \{ q_{G} \}) \mapsto \lambda_{G}(m) \).
- \( \triangledown_{S} = \{(x, q_{S}) \} \cup \{(q_{S}, m) \mid G \in S, q_{G} \vdash_{G} m \} \cup \bigcup_{G \in S}(\triangledown \setminus \{(x, q_{G}) \} \setminus \{(q_{G}, m) \mid q_{G} \vdash_{G} m \}) \).
- \( P_{S} = \text{pref}(\{(q_{S}, s) \mid \exists G \in S, q_{G} \in P_{G} \}) \), where moves \( m \) in \( s \) with \( q_{G} \vdash_{G} m \) are justified by \( q_{S} \).

That is, the parallel union construction \( S \) forms a predicative game from a set of predicative games by “unifying the first moves”.

**Remark.** The parallel union \( S \) appears similar to the product \& defined below. However, there is a definitive difference: Player always chooses the component game to play in the former, while Opponent does it in the latter.

It is easy to see that:

**Lemma 3.3.5** (Well-defined parallel union). For any \( k \in \mathbb{Z}_{\geq 2} \) and \( S \subseteq PG_{\leq k} \), the parallel union \( S \) is a well-defined \( \sup(\{ T(G) \mid G \in S \}) \)-predicative game that satisfies \( \text{gs}(S) = \bigcup_{G \in S} \text{gs}(G) \).

**Proof.** Clear from the definition.

Another construction is similar but on generalized strategies:

**Definition 3.3.6** (Predicative union of generalized strategies). Given an integer \( k \in \mathbb{Z}_{\geq 1} \) and a set \( S \subseteq GS_{\leq k} \) of generalized strategies with type \( \leq k \), we define the predicative game \( S \), called the **predicative union** of \( S \), as follows:

- \( M_{S} = \{ q_{S} \} + \sum_{S \in S}(N(S) \cup M_{S}) \), where \( q_{S} = [0]_{0} \).
- \( \lambda_{S} : q_{S} \mapsto G, (m \in M_{S}) \mapsto \lambda_{S}(m) \).
- \( \triangledown_{S} = \{(x, q_{S}) \} \cup \{(q_{S}, N) \mid S \in S \} \cup \{(N(S), m) \mid S \in S, \vdash_{S} m \} \cup \bigcup_{S \in S}(\triangledown_{S} \setminus \{(x, N) \} \setminus \{(N, m) \mid N \vdash_{S} m \}) \).
- \( P_{S} = \text{pref}(\{(q_{S}, N(S), s) \mid S \in S, s \in P_{S} \}) \), where \( N(S) \) is justified by \( q_{S} \) and the moves in \( s \) that are initial in \( S \) are justified by \( N(S) \).

Then it is immediate that:

**Lemma 3.3.7** (Well-defined predicative union). For any \( k \in \mathbb{Z}_{\geq 1} \) and \( S \subseteq GS_{\leq k} \), the predicative union \( S \) is a well-defined \( \sup(\{ T(S) \mid S \in S \}) + 1 \)-predicative game that satisfies \( \text{gs}(S) = S \). Moreover, any predicative game \( G \in PG \) is the predicative union of its strategies, i.e., \( G = \bigcup \text{gs}(G) \).

**Proof.** Clear from the definition.
The lemma enables us to show that $\mathcal{PG}_{k} \neq \emptyset$ for all $k \in \mathbb{Z}_{\geq 2}$ in a concise way: We define a 2-predicative game $I_{2} \overset{\text{df}}{=} \{I_{1}\}$, where $I_{1} \overset{\text{df}}{=} I$ is the empty game; we then inductively define a $(k+1)$-predicative game $I_{k+1} \overset{\text{df}}{=} \{I_{k}\}$ for each $k \in \mathbb{Z}_{\geq 2}$. As a consequence, we have $I_{k} \in \mathcal{PG}_{k}$ for all $k \in \mathbb{Z}_{\geq 2}$. Also, we have $I_{j} : I_{j+1}$ for all $j \in \mathbb{Z}_{\geq 1}$, establishing another fact that $\mathcal{GS}_{j} \neq \emptyset$ for all $j \in \mathbb{Z}_{\geq 1}$.

Now, let us sketch how the “protocol” of predicative games works to interpret intuitionistic type theory. A distinguishing feature of the type theory is dependent types, i.e., families $(B(a) : \mathcal{U})_{a : A}$ of types $B(a)$ indexed by terms $a$ on a type $A$. Moreover, we may construct a dependent function type $\prod_{a : A} B(a)$ whose terms represent functions $f : A \to \bigcup_{a : A} B(a)$ that satisfy $f(x) : B(x)$ for all $x : A$. Note that a dependent function type $\prod_{a : A} B(a)$ is a generalization of a function type $A \to B$ where $B$ is a “constant” dependent type, i.e., $B(a) = B$ for all $a : A$.

It is then a challenge to interpret this structure in game semantics: Omitting the semantic bracket $[\cdot]$, an interpretation of the dependent function type $\prod_{a : A} B(a)$ should be a generalization of implication $A \to B = \uparrow A \to B$ of games, where $B$ may depend on the strategy on $A$ which Opponent chooses to play. Our solution is to require the “protocol” of predicative games: We shall interpret the dependent function type by the subgame of the implication $A \to \{B(\sigma) | \sigma : A\}$, in which we require the “type-matching” condition on strategies $\tau$, i.e., $\tau \circ \sigma_{1} : B(\sigma)$ for all $\sigma : A$. Now, the “protocol” becomes something like $q_{B} : q_{A} \cdot \mathcal{N}(\sigma) \cdot \mathcal{N}(\tau \circ \sigma_{1})$, and then a play in the specified implication $\sigma \to \tau \circ \sigma_{1}$ follows.

This nicely captures the phenomenon of dependent function types, but imposes another challenge: The implication $A \to \{B(\sigma) | \sigma : A\}$ no longer has the “protocol” because the second move $q_{A}$ is not the name of the game to follow. Even if we somehow enforce the “protocol”, then it seems that we would lose the initial questions and answers $q_{B} : q_{A} \cdot \mathcal{N}(\sigma) \cdot \mathcal{N}(\tau \circ \sigma_{1})$ to determine the game $\sigma \to B(\sigma)$ to play. We address this problem in Section 4 and in fact interpret dependent function types in Section 5.2.

### 3.4 Subgame Relation

Next, we generalize the subgame relation on games defined in Section 2.1.6 to define an appropriate one on predicative games.

**Definition 3.4.1** (Subgame relation on predicative games). A **subgame** of a $k$-predicative game $G$, $k \in \mathbb{Z}_{\geq 2}$, is a $k$-predicative game $S$ that satisfies $\text{gs}(S) \subseteq \text{gs}(G)$. In this case, we write $S \leq G$. Moreover, we write $\text{sub}(G)$ for the set of all subgames of $G$.

That is, $S \leq G \overset{\text{df}}{=} T(S) = T(G) \cap \text{gs}(S) \subseteq \text{gs}(G)$. Thus, in particular, the subgame relation $\leq$ forms a partial order on the set $\mathcal{PG}_{k}$ for each $k \in \mathbb{Z}_{\geq 2}$. Also, because of the generalization which predicative games make, the subgame relation between predicative games is finer than the subgame relation between MC-games (see Definition 2.1.9).

### 3.5 Universe Games

Now, we interpret universes by certain predicative games, which should be called **universe games**. Note that we are interested in a predicative type theory with a hierarchy of universes; so we shall construct the corresponding hierarchy of the universe games.

**Definition 3.5.1** (Universe games). For each natural number $k \in \mathbb{N}$, the $k$-th **universe game** $U_{k}$ is defined by $U_{k} \overset{\text{df}}{=} \mathcal{PG}_{k+2}$. A **universe game** refers to the $k$-th universe game $U_{k}$ for some $k \in \mathbb{N}$, and it is often abbreviated as $U$. 
Of course, we have:

- **Corollary 3.5.2** (Predicativity of universe games). For each \( k \in \mathbb{N} \), the \( k \)th universe game \( U_k \) is a well-defined \((k + 3)\)-predicative game.

*Proof.* Let \( k \in \mathbb{N} \) be fixed. By Lemma 3.3.7 it suffices to show that \( \mathcal{P} \mathcal{G}_{\leq k+2} \subseteq \mathcal{G} \mathcal{S}_{\leq k+2} \) and \( \exists S \in \mathcal{P} \mathcal{G}_{\leq k+2} \cdot T(S) = k + 2 \). But we have \( G : \mathcal{P} \{ G \} \) for all \( G \in \mathcal{P} \mathcal{G}_{\leq k+2} \) again by Lemma 3.3.7, showing the inclusion \( \mathcal{P} \mathcal{G}_{\leq k+2} \subseteq \mathcal{G} \mathcal{S}_{\leq k+2} \). And as we have seen before, we have \( I_{k+2} \in \mathcal{P} \mathcal{G}_{\leq k+2} \).

By Lemma 3.3.7 \( G \in \mathcal{P} \mathcal{G}_{\leq k+2} \Leftrightarrow G : U_k \) for all \( k \in \mathbb{N} \). Thus, as intended, the \( k \)th universe game \( U_k \) is the "universe" of all \( i \)-predicative games with \( 2 \leq i \leq k + 2 \). Intuitively, a play of a universe game \( U \) starts with the Opponent’s question \( q_U \text{ "What is your game?"} \), and Player answers it by the name of a predicative game such as \( \mathcal{N}(G) \), meaning "It is the game \( G \)!", and then a play in \( G \) follows.

As a consequence, we have \( U_i : U_j \) for all \( i, j \in \mathbb{N} \) with \( i < j \). Thus, we obtain a hierarchy of the universe games: \( U_0 : U_1 : U_2 \ldots \)
4 The Category of Predicative Games and Generalized Strategies

In this section, we first modify the constructions on games and strategies of [McC98] (see Sections 2.1.5, 2.1.7) in such a way that the new ones preserve the predicativity of games, and then define the category $\mathcal{PG}$ of predicative games and generalized strategies. As explained earlier, we cannot simply adopt the usual constructions, because, e.g., the linear implication of two predicative games is not predicative (as the second move would not be the name of a game to follow); thus, we need appropriate modifications.

First, for convenience, we employ:

◮ Notation. We write $\otimes_{i \in I} G_i$ either for the tensor product $G_1 \otimes G_2$, linear implication $G_1 \to G_2$, product $G_1 \& G_2$, exponential $!G_1$, or composition $G_2 \circ G_1$ of predicative games $G_i, i \in I$, which we shall define below, where $I$ is an appropriate “index set”. Similarly, we write $\&_{j \in J} S_j$ either for the tensor product $S_1 \otimes S_2$, composition $S_2 \circ S_1$, paring $S_1 \& S_2$, or promotion $!S_1$ of generalized strategies $S_i, j \in J$, which we will define below.

◮ Remark. For a uniform treatment, we write $\&$ and $!$ for paring and promotion of strategies, respectively. In fact, paring and promotion are essentially the same operations with product and exponential of games, respectively, in the usual game semantics; see Sections 2.1.5, 2.1.7. As we shall see, it is the case for predicative games as well.

◮ Convention. Whenever we say “a composition $T \circ S$ of games”, the games $S, T$ are always assumed to be “composable”, i.e., $S \leq A \circ B$ and $T \leq B \circ C$ for some games $A, B, C$.15

4.1 Constructions on Predicative Games

In this section, we define constructions on predicative games. As a preparation, observe the following:

◮ Lemma 4.1.1 (Type increment lemma). Every predicative game $G$ induces another predicative game $G^{+1}$ with $T(G^{+1}) = T(G) + 1$ and $G : G^{+1}$ defined by $G^{+1} \overset{df}{=} \{G\}$.

Proof. By Lemma 3.3.7.

As a consequence, we may induce a predicative game $G^{+n}$ with $T(G^{+n}) = T(G) + n$ for any predicative game $G$ and natural number $n$: $G^{+n}$ is constructed from $G$ by the $n$-time iteration of the operation $(\_)^{+1}$ in Lemma 4.1.1. Note that $G^{+n}$ is a trivial modification of $G$ that just increases the type of $G$ by $n$.

Also, the lemma enables us to generalize the phenomenon of the existing games and strategies that a strategy $\sigma : G$ is a particular kind of subgame $\sigma \subseteq G$ (see Section 2.1.6).

◮ Proposition 4.1.2 (Generalized strategies as subgames). A generalized strategy $\sigma : G$ induces a subgame $\sigma^{+1} \subseteq G$.

Proof. Immediate from the definition.

A main challenge in this section is how to define linear implication inductively; the other constructions have obvious inductive structures. For this, we need the following:

15This is a generalization of the following fact: For MC-games and innocent strategies, we may generalize composition of strategies to the one on games of this kind (i.e., subgames of composable linear implications).

16It is also similar to the phenomenon in sets that an element $x \in X$ induces a subset $\{x\} \subseteq X$.24
Definition 4.1.3 (Families of PLIs). Let $A, B$ be predicative games with the same type. A family of point-wise linear implications (or family of PLIs) from $A$ to $B$ is a pair $(f, \phi_{\sigma}|_{\sigma:A})$ of a function $f \in \text{gs}(B)^{\text{gs}(A)}$, called the extensional collapse, and a collection $(\phi_{\sigma} \leq \sigma \rightarrow f(\sigma))|_{\sigma:A}$ of subgames $\phi_{\sigma}$, called the point-wise linear implications (or PLIs). If $A, B$ are $2$-predicative, then we additionally require that $\phi_{\sigma}$ is deterministic on even-length positions for all $\sigma : A$.

Of course this definition is applicable only for $2$-predicative games at the moment (PLIs between $2$-predicative games are defined via the usual linear implication of games); PLIs for other cases will be defined as soon as we define linear implication inductively below.

Notation. A family of PLIs from $A$ to $B$ is usually represented just by its PLIs $(\phi_{\sigma})|_{\sigma:A}$ with its extensional collapse, usually written $\pi_{\phi}$, implicit.

Now, we are ready to define constructions on predicative games:

Definition 4.1.4 (Constructions on predicative games). For any family $(G_i)_{i \in I}$ of predicative games, the predicative game $\otimes_{i \in I} G_i$ is inductively defined as follows:

1. First, by the operation $(\_)^{+1}$ in Lemma 4.1.1 we may assume that the predicative games $G_i, i \in I$, are all of $\sup\{|T(G_i)| i \in I\}$-type.

2. We define $\otimes_{i \in I} G_i \overset{df}{=} \{\otimes_{i \in I} \sigma_i \mid \forall i \in I. \sigma_i : G_i\}$ if $\otimes$ is either tensor product, product, exponential, or composition; and define:

   $$G_1 \rightarrow G_2 \overset{df}{=} \{\sigma \mid \exists \sigma_1 : G_1 (\exists \sigma_2 : G_2. \sigma = \sigma_1 \bullet \sigma_2)\}
   \quad \text{if } \bullet \text{ is linear implication, where } \exists \text{ is the obvious generalization of binary product, and the}
   \quad \text{composition of products } \phi \overset{df}{=} \{\phi_{\sigma_1} | \sigma_1 : G_1\} : G_1 \rightarrow G_2, \psi \overset{df}{=} \{\psi_{\sigma_2} | \sigma_2 : G_2\} : G_2 \rightarrow G_3
   \quad \text{is defined by } \psi \circ \phi \overset{df}{=} \{\psi_{\sigma_2} \circ \phi_{\sigma_1} | \sigma_1 : G_1\} : G_1 \rightarrow G_3,
   \quad \text{where note that } \pi_{\phi \circ \psi} = \pi_{\psi \circ \phi}.
   \quad \text{For the base case, we apply the usual constructions (in Sections 2.1.5 2.1.7).}

Notation. Let $G_1, G_2$ be predicative games. A generalized strategy $\{\phi_{\sigma_1} | \sigma_1 : G_1\}$ on a linear implication $G_1 \rightarrow G_2$ is usually written $\phi$; in other words, a generalized strategy $\phi : G_1 \rightarrow G_2$ is a product $\phi = \{\phi_{\sigma_1} | \sigma_1 : G_1\}$ of subgames $(\phi_{\sigma_1} \leq \sigma_1 \rightarrow \pi_{\sigma}(\sigma_1))|_{\sigma_1:G_1}$, where how each strategy $\phi_{\sigma_1}$ plays on $G_2$ is determined by a specified function $\pi_{\phi} \in \text{gs}(G_2)^{\text{gs}(G_1)}$.

That is, the constructions on predicative games are essentially the usual constructions on games (in Section 2.1.5), augmented with a systematic operation on the questions and answers about the name of games that preserves predicativity. In fact, if we take $2$-predicative games that correspond to MC-games (in the way we did in Section 3.2), then it is straightforward to see that our constructions are essentially the same as the constructions on MC-games (in Section 2.1.5).

We now establish an important fact:

Theorem 4.1.5 (Well-defined constructions on predicative games). The class of predicative games is closed under all the constructions defined in Definition 4.1.4 except for tensor product and exponential which just do not preserve well-openedness.

Proof. For tensor product, product, exponential, and linear implication, it is straightforward to show the theorem by a simple induction on $\sup\{|T(G_i) | i \in I\}$ with Lemma 3.5.7. It remains to consider composition; the following lemma handles it.

Lemma 4.1.6 (Composition on subgames). Let $S \leq A \rightarrow B, T \leq B \rightarrow C$ be any predicative games. Then we have the subgame relation $T \circ S \leq A \rightarrow C$. 

25
Proof of the lemma. Assume that $S, T$ are both $k$-predicative by Lemma 4.1.1. We show the claim by induction on $k$. The base case $k = 0$ is easier than and similar to the inductive case, so we omit it here. For the inductive case $k + 1$, recall that:

$$S \leq A \rightarrow B \iff \text{gs}(S) \subseteq \text{gs}(A \rightarrow B) = \{ \phi \mid (\phi_\sigma)_{\sigma : A} \text{ is a family of PLIs from } A \text{ to } B \}$$

$$T \leq B \rightarrow C \iff \text{gs}(T) \subseteq \text{gs}(B \rightarrow C) = \{ \psi \mid (\psi_\tau)_{\tau : B} \text{ is a family of PLIs from } B \text{ to } C \}$$

Hence, by the definition,

$$T \circ S = \{ \phi \circ \phi \mid \phi \in \text{gs}(S), \psi \in \text{gs}(T) \}$$

$$= \{ \&\{\psi_\tau \circ \phi_\sigma \mid \sigma : A\} \mid \phi \in \text{gs}(S), \psi \in \text{gs}(T) \}$$

where $\forall \sigma : A. \psi_\tau \circ \phi_\sigma \leq \sigma \rightarrow (\pi_\psi \circ \pi_\phi)(\sigma) = \sigma \rightarrow \pi_\psi \circ \pi_\phi(\sigma)$ by the induction hypothesis. Therefore we may conclude that $\text{gs}(T \circ S) \subseteq \text{gs}(A \rightarrow C)$, whence $T \circ S \leq A \rightarrow C$.

This completes the proof of the theorem.

Note in particular that when we focus on 2-predicative games that correspond to MC-games (see Section 3.2), the tensor product, product, exponential, and linear implication on predicative games are essentially the same as the corresponding operations on MC-games (in Section 2.1.5). This (at least partially) justifies our constructions on predicative games.

Let us see a simple example.

Example 4.1.7. In MC-games and innocent strategies, we have a game $N_{mc}$ of natural numbers, defined by $M_{mc} \equiv \{ q \} + \mathbb{N}; \lambda_{mc} : q \mapsto \mathbb{Q}, (n \in \mathbb{N}) \mapsto \mathbb{P}A; \tau_{mc} \equiv \left\{ \{ *, q \} \cup \{ (q, n) \mid n \in \mathbb{N} \} \right\}; P_{mc} \equiv \text{pref}(\{ (q, m) \mid m \in \mathbb{N} \})$. Let us write $n$ for the strategy $\text{pref}(\{ (q, n) \mid n \in \mathbb{N} \})$ for each $n \in \mathbb{N}$. The corresponding 2-predicative game $N$ consists of: $M_N \equiv \{ qn \} + \{ N(n) \mid n \in \mathbb{N} \} + \{ q_{\mathbb{N}} \mid n \in \mathbb{N} \} + (\mathbb{N} \times \{ 0 \}); \lambda_N : qn \mapsto \mathbb{Q}, N(n) \mapsto \mathbb{P}A, q_{\mathbb{N}} \mapsto \mathbb{Q}, ((n|_0 \in \mathbb{N} \times \{ 0 \}) \mapsto \mathbb{P}A; \tau_N \equiv \left\{ \{ *, qn \} \cup \{ (qn, N(n)) \mid n \in \mathbb{N} \} \cup \{ (N(n), \mathbb{N}) \mid n \in \mathbb{N} \} \cup \{ (\mathbb{N}, \mathbb{N}) \mid n \in \mathbb{N} \} \cup \{ (\mathbb{N}, \mathbb{N}) | n \mid \in N \} \right\}; P_N \equiv \text{pref}(\{ (qn, N(n)), q_{\mathbb{N}} \mid n \mid \in N \})$. Moreover, typical plays in the linear implication $N \rightarrow N$ are of the form $qn \rightarrow N.N(\phi),s$, where $s \in \phi_\tau : \tau \rightarrow \pi_\phi(\tau)$ for some $n \in \mathbb{N}$. That is, Opponent first asks a question $qn \rightarrow N$ about a strategy to play, and Player declares a (strict) strategy $\phi = \&\{ \phi_n \mid n \in \mathbb{N} \}$, then a play in the usual implication $N_{mc} \rightarrow N_{mc}$ follows, in which Opponent first has to (implicitly) select a strategy $n$, and then a play by $\phi_n$ follows. It is clear that the linear implication $N \rightarrow N$ is essentially the same as the usual one $N_{mc} \rightarrow N_{mc}$. Also, this example illustrates the point that our linear implication preserves the predicativity of games.

### 4.2 Constructions on Generalized Strategies

We may apply the same idea to define constructions on generalized strategies.

Definition 4.2.1 (Constructions on generalized strategies). For a family $(S_j)_{j \in J}$ of generalized strategies, the generalized strategy $\bigoplus_{j \in J} S_j$ is inductively defined as follows:

1. First, by the operation $\_ \_^{-1}$ in Lemma 4.1.1, we may assume that the generalized strategies $S_j, j \in J$, are all of sup$(\{ T(S_j) \mid j \in J \})$-type.

---

17 Here, we use the established fact in MC-games that composition of innocent strategies is well-defined; see [McC98] for the details.
2. If the generalized strategies \( S_j, j \in J \), are all strict, and \( \intercal \) is not composition, then we define \( \intercal_{j \in J} S_j \overset{\text{df}}{=} \intercal_{j \in J} S_j \), where \( \intercal \) is the corresponding construction on strategies defined in Section 2.1.7.

3. Otherwise, we define \( \intercal_{j \in J} S_j \overset{\text{df}}{=} \{ \intercal_{j \in J} \tau_j \mid \forall j \in J, \tau_j : S_j \} \) if \( \intercal \) is either tensor product, paring, or promotion; and define \( \psi \circ \phi \overset{\text{df}}{=} \& \{ \psi_{\pi_{\sigma}(\sigma)} \circ \phi_{\sigma} \mid \sigma : A \} \) if \( \intercal \) is composition and \( S_1 = \phi : A \rightarrow B, S_2 = \psi : B \rightarrow C \) for some predicative games \( A, B, C \), where \( \psi_{\pi_{\sigma}(\sigma)} \circ \phi_{\sigma} \) is the usual composition defined in Section 2.1.7 for the base case or the composition on predicative games (see Definition 4.1.4) for the inductive case.

Now, we prove one of the main theorems of the present paper:

\\textbf{Theorem 4.2.2 (Well-defined constructions on generalized strategies).} The class of generalized strategies is closed under all the constructions defined in Definition 4.2.1, except for tensor product and promotion which just do not preserve well-openedness. Moreover,

1. The tensor product, paring, and promotion on non-strict strategies coincide with the tensor product, product, and exponential on predicative games, respectively.

2. If \( \sigma_1 : D \rightarrow A_1, \sigma_2 : D \rightarrow A_2, \tau : A \rightarrow B, \kappa : B \rightarrow C \), \( \phi : !A \rightarrow B, \psi : !B \rightarrow C \), then
   \( \sigma_1 \& \sigma_2 : D \rightarrow A_1 \& A_2, \kappa \circ \tau : A \rightarrow C, \psi \circ \phi : !A \rightarrow C \).

\textit{Proof.} It is straightforward to show that each construction is well-defined by induction on \( \mathsf{sup}(\{ T(S_j) \mid j \in J \}) \) with Lemmata 3.3.7, 4.1.6, where the case for strict strategies is already established in the literature (see [McC98] for the proofs). The clauses 1, 2 are obvious by the definition, where for \( \sigma_1 \& \sigma_2 : D \rightarrow A_1 \& A_2 \) note that we have a correspondence \( D \rightarrow A_1 \& A_2 \cong (D \rightarrow A_1) \& (D \rightarrow A_2) \).

\\textbf{Example 4.2.3.} If we have \( A : U_i \) and \( B : U_j \), then, \( A \rightarrow B, A \& B : U_{k} \), where \( k \overset{\text{df}}{=} \max(i, j) \).

We often represent this phenomenon rather casually by \( A : U \& B : U \Rightarrow A \rightarrow B, A \& B : U \).

We have defined various constructions on generalized strategies, but we do not have any "atomic" strategies yet. In [AJM00, McC98], there are two such atomic strategies: The \textbf{copy-cat strategies} \( c_{G} : G \rightarrow G \), and the \textbf{derelictions} \( \text{det}_{G} : !G \rightarrow G \) (see Section 2.1.7). We now generalize these two strategies.

\\textbf{Definition 4.2.4 (Generalized copy-cat strategies).} The \textbf{(generalized) copy-cat strategy} \( c_{G} : G \rightarrow G \) on a \( k \)-predicative game \( G, k \in \mathbb{Z}_{\geq 2} \), is inductively defined as follows:

- If \( G \) is 2-predicative, then we define \( c_{G} \overset{\text{df}}{=} \& \{ c_{\sigma} \mid \sigma : G \} \), where \( c_{\sigma} : \sigma \rightarrow \sigma \) is the usual one defined in Definition 2.1.20 (for example, the identity function on \( \mathsf{gs}(G) \)).

- If \( G \) is \( (k + 1) \)-predicative, then we define \( c_{G} \overset{\text{df}}{=} \& \{ c_{\sigma+1} \mid \sigma : G \} \), where \( c_{\sigma} \overset{\text{df}}{=} \mathsf{id}_{\mathsf{gs}(G)} \) is defined as above.

The idea is simple: The copy-cat strategy \( c_{G} \) on a \( k \)-predicative game \( G \) just waits until a play has reached at a 2-predicative game \( G' \), and then it plays as the usual copy-cat strategy \( c_{G'} \).

\footnote{For this, we think of strict strategies as innocent strategies on themselves (as games) by Lemma 2.1.19.}

\footnote{More precisely, we inductively apply a slight modification of \( \sigma_1 \& \sigma_2 \) as in Definition 2.1.20.}
Lemma 4.2.5 (Well-defined copy-cat strategies). For any predicative game \( G \), the copy-cat strategy \( \text{cp}_G \) is a well-defined strategy on \( G \rightarrow G \) that satisfies the unit law (with respect to composition \( \circ \)).

Proof. First, for any predicative game \( G \), it is straightforward to see that the copy-cat strategy \( \text{cp}_G \) is a well-defined generalized strategy on \( G \rightarrow G \) by induction on the type of \( G \).

Next, let \( A \rightarrow B \) be a linear implication. We show, by induction on the type of the linear implication, that \( \kappa \circ \text{cp}_A = \kappa \) and \( \text{cp}_B \circ \kappa = \kappa \) for all \( \kappa : A \rightarrow B \), and \( S \circ \text{cp}_A^+ = S \) and \( \text{cp}_B^+ \circ S = S \) for all \( S \leq A \rightarrow B \). For the base case in which \( T(\kappa) = 1 \), we have:

\[
\begin{align*}
\kappa \circ \text{cp}_A &= \kappa \circ \{ \kappa_\sigma | \sigma : A \} \circ \{ \text{cp}_\sigma | \sigma : A \} = \{ \kappa \circ \text{cp}_\sigma | \sigma : A \} = \kappa \\
\text{cp}_B \circ \kappa &= \{ \text{cp}_\sigma | \tau : B \} \circ \kappa \circ \{ \kappa_\sigma | \sigma : A \} = \{ \text{cp}_\sigma | \kappa_\sigma | \sigma : A \} = \kappa
\end{align*}
\]

showing that the unit law holds (where we use the fact that the usual copy-cat strategies in MC-games satisfy the unit law; see [McC98]). Also, it follows that, for any subgame \( S \leq A \rightarrow B \), we have:

\[
\begin{align*}
S \circ \text{cp}_A^+ &= \{ \kappa \circ \text{cp}_A | \kappa : S \} = \{ \kappa | \kappa : S \} = S \\
\text{cp}_B^+ \circ S &= \{ \text{cp}_B \circ \kappa | \kappa : S \} = \{ \kappa | \kappa : S \} = S
\end{align*}
\]

by what we have just shown above and Lemma 3.3.7. For the inductive case in which \( T(\kappa) = k + 1 \), we have:

\[
\begin{align*}
\kappa \circ \text{cp}_A &= \{ \kappa_\sigma | \sigma : A \} \circ \{ \text{cp}_A^+ | A : A \} = \{ \kappa \circ \text{cp}_A^+ | A : A \} = \{ \kappa_\sigma | \sigma : A \} = \kappa \\
\text{cp}_B \circ \kappa &= \{ \text{cp}_B^+ | B : B \} \circ \sigma \circ \{ \kappa_\sigma | \sigma : A \} = \{ \text{cp}_B^+ \circ \kappa_\sigma | \sigma : A \} = \{ \kappa_\sigma | \sigma : A \} = \kappa
\end{align*}
\]

by the induction hypothesis, showing that the unit law holds. Again, it follows that, for any subgame \( S \leq A \rightarrow B \), we have:

\[
\begin{align*}
S \circ \text{cp}_A^+ &= \{ \kappa \circ \text{cp}_A | \kappa : S \} = \{ \kappa | \kappa : S \} = S \\
\text{cp}_B^+ \circ S &= \{ \text{cp}_B \circ \kappa | \kappa : S \} = \{ \kappa | \kappa : S \} = S
\end{align*}
\]

again by what we have just shown above and Lemma 3.3.7 which completes the proof. 

In a completely analogous way, we may generalize derelictions:

Definition 4.2.6 (Generalized derelictions). The (generalized) dereliction \( \text{der}_G : !G \rightarrow G \) on a \( k \)-predicative game \( G \), \( k \in \mathbb{Z}_{\geq 2} \), is inductively defined as follows:

- If \( G \) is \( 2 \)-predicative, then we define \( \text{der}_G \) as \( \{ \text{der}_\sigma | \sigma : G \} \), where \( \text{der}_\sigma : \sigma \rightarrow G \) is the usual one defined in Definition 2.1.26 and \( \pi_{\text{der}} \in \text{gs}(G)^{\text{gs}(G)} \) is defined to be the function such that \( \pi_{\text{der}} \circ \sigma \) for all \( \sigma : G \).

- If \( G \) is \( (k+1) \)-predicative, then we define \( \text{der}_G \) as \( \{ \text{der}_\sigma^+ | \sigma : G \} \), where \( \pi_{\text{der}} \in \text{gs}(G)^{\text{gs}(G)} \) is defined as above.

By the same way as the case of copy-cat strategies, we may establish:

Lemma 4.2.7 (Well-defined derelictions). For any predicative game \( G \), the dereliction \( \text{der}_G \) is a well-defined strategy on \( G \rightarrow G \) that satisfies the unit law (with respect to “promotion plus composition”: see Definition 3.3.1 below for this notation).
4.3 The Category of Predicative Games and Generalized Strategies

Based on constructions on predicative games and generalized strategies defined in the previous sections, we now proceed to define a category of predicative games and generalized strategies.

Definition 4.3.1 (The category \( \mathcal{PG} \)). We define the category \( \mathcal{PG} \) of predicative games and generalized strategies as follows:

- Objects are predicative games.
- A morphism \( A \to B \) is a generalized strategy \( S : !A \to B \).
- The composition of morphisms \( S : !A \to B \) and \( T : !B \to C \) is \( T \circ S : !A \to C \).
- The identity \( \text{id}_A \) on each object \( A \) is the generalized dereliction \( \text{der}_A : !A \to A \).

We usually write \( A \to B \) for \( !A \to B \).

It is immediate to see that:

Corollary 4.3.2 (Well-defined \( \mathcal{PG} \)). The structure \( \mathcal{PG} \) forms a well-defined category.

Proof. By Theorems 4.1.5, 4.2.2 and Lemma 4.2.7, it remains to show the associativity of the composition. But it immediately follows from the associativity of the usual composition in Section 2.1.7 (see [McC98] for a proof) by induction on the type of generalized strategies.

Convention. From now on, we often call predicative games and generalized strategies just games and strategies, respectively.
5 Game-theoretic Interpretation of ETT

We now propose in this section an interpretation of the extensional variant of intuitionistic type theory (ETT) by predicative games and generalized strategies. Our approach is based on a categorical model of the type theory, called categories with families (or CwFs for short) [Dyb96], because it is in general easier to show that a structure is an instance of a CwF than to directly establish that it is a model of the type theory.

We first define additional constructions on games, which serve as preliminary notions, and then we define our game-theoretic CwF $\mathcal{EPG}$ for ETT, and equip it with semantic type formers [Hof97] which interpret specific types such as $\prod$, $\Sigma$, and Id-types as well as universes.

5.1 Dependent Games and Dependent Union

In the type theory, a dependent type $B$ over a type $A$ is a family $\{B(a) \mid a : A\}$ of types $B(a)$ indexed by terms $a$ on a fixed type $A$. Hence, it is natural to define:

|$\begin{align*}
\text{◮ Definition 5.1.1 (Dependent games).} \\
\text{A dependent game over a predicative game } A \text{ is a family } B = \{B\sigma : U \mid \sigma : A\} \text{ of predicative games } B\sigma \text{ indexed by strategies } \sigma \text{ on } A. \\
\text{◭}
\end{align*}$|

Note that we can identify a dependent game $B$ over $A$ with the strategy $B : A \to U$ that plays exclusively in $U$ (after the "protocol") in the obvious way.

To interpret various types, in which dependent types are involved, the following construction plays an important role:

|$\begin{align*}
\text{◮ Definition 5.1.2 (Dependent union).} \\
\text{Given a dependent game } B \text{ over a game } A, \text{ we define its dependent union } \uplus B \text{ by } \uplus B \overset{df}{=} \int \{B\sigma \mid \sigma : A\}. \\
\text{◭}
\end{align*}$|

By Lemma 3.3.5, the construction of dependent union is well-defined, where a universe game $U$ serves as an "upper bound" of the types of $B\sigma$. With this construction, it is now clear that a predicative game is a generalization of a game that can be a "family of games". We shall see concrete examples shortly.

5.2 Dependent Function Space

We proceed to define a game-theoretic structure to interpret dependent function types (also called dependent product types or $\prod$-types) $\prod_{x : A} B(a)$, where $A$ is a type and $B : A \to U$ is a dependent type, in intuitionistic type theory.

|$\begin{align*}
\text{◮ Definition 5.2.1 (Dependent function space).} \\
\text{The dependent function space } \hat{\prod}(A, B) \text{ of a dependent game } B \text{ over a game } A \text{ is the subgame of the implication } A \to \uplus B \text{ whose strategies } \tau \text{ satisfy } \forall \sigma : A. \tau \bullet \sigma : B\sigma. \\
\text{◭}
\end{align*}$|

The idea is simple: It represents the type of (set-theoretic) functions $f : A \to \bigcup_{x \in A} B(x)$ that satisfies $f(a) \in B(a)$ for all $a \in A$. Note that when $B$ is the "constant" dependent game, i.e., $B\sigma = B$ for all $\sigma : A$, the dependent function space $\hat{\prod}(A, B)$ coincides with the implication $A \to B$ as expected.

However, we will have to handle the case where $A$ is a dependent game; so this definition is not general enough. In terms of the type theory, we can interpret the rule

$$\Gamma, x : A \vdash B(x) \text{ type } \Rightarrow \Gamma \vdash \prod_{x : A} B(x) \text{ type}$$
only when $\Gamma = \Diamond$ (the empty context) at the moment. This is why we use the symbol $\hat{\prod}$ here; we shall define a more general construction $\prod$ of dependent function space shortly.

30
for the strategy

We are now ready to define an instance of a CwF by predicative games and generalized strategies. For the details, see [Hof97].

Semantic counterparts in a CwF for all the contexts, types, and terms in the type theory, and every judgemental equality is preserved as the corresponding equation between the interpretations; for the details, see [Hof97].
Definition 5.5.1 (CwFs [Dyb96, Hof97]). A category with families (or a CwF for short) is a structure \( \mathcal{C} = \langle \mathcal{C}, \text{Ty}, \text{Tm}, \_\{\_\}, \_\{\_\}, \_\{\_\}, \_\{\_\}, \_\{\_\}, \text{Ty-Id.}, \text{Tm-Comp.} \rangle \), where:

- \( \mathcal{C} \) is a category.
- \( \text{Ty} \) assigns, to each object \( \Gamma \in \mathcal{C} \), a set \( \text{Ty}(\Gamma) \), called the set of types in the context \( \Gamma \).
- \( \text{Tm} \) assigns, to each pair of an object \( \Gamma \in \mathcal{C} \) and a type \( A \in \text{Ty}(\Gamma) \), a set \( \text{Tm}(\Gamma, A) \), called the set of terms of type \( A \) in the context \( \Gamma \).
- For each morphism \( \phi : \Delta \to \Gamma \) in \( \mathcal{C} \), \( \_\{\_\} \) induces a function \( \_\{\_\} : \text{Ty}(\Gamma) \to \text{Ty}(\Delta) \) and a family of functions \( \langle \_\{\_\} \rangle_A : \text{Tm}(\Gamma, A) \to \text{Tm}(\Delta, A\{\_\}) \rangle_{A \in \text{Ty}(\Gamma)} \).
- \( \_\{\_\} \) assigns, to each pair 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 \( A \).
- \( \_\{\_\} \) assigns, to each pair of a context \( \Gamma \in \mathcal{C} \) and a type \( A \in \text{Ty}(\Gamma) \) with a morphism \( p(A) : \Gamma.A \to \Gamma \) in \( \mathcal{C} \), called the first projection associated to \( A \).
- \( \_\{\_\} \) assigns, to each pair of a context \( \Gamma \in \mathcal{C} \) and a type \( A \in \text{Ty}(\Gamma) \) with a term \( v_A \in \text{Tm}(\Gamma.A, A\{p(A)) \} \), called the second projection associated to \( A \).
- \( \_\{\_\} \) assigns, to each triple of a morphism \( \phi : \Delta \to \Gamma \in \mathcal{C} \), a type \( A \in \text{Ty}(\Gamma) \), and a term \( \tau \in \text{Tm}(\Delta, A\{\_\}) \), a morphism \( \langle \_\{\_\} \rangle_A : \Delta \to \Gamma.A \in \mathcal{C} \), called the extension of \( \phi \) by \( \tau \).

Moreover, it is required to satisfy the following axioms:

- **Ty-Id.** For each pair of a context \( \Gamma \in \mathcal{C} \) and a type \( A \in \text{Ty}(\Gamma) \), we have \( A\{\text{id}_\Gamma\} = A \).
- **Ty-Comp.** Additionally, for any composable morphisms \( \phi : \Delta \to \Gamma \), \( \psi : \Theta \to \Delta \) in \( \mathcal{C} \), we have \( A\{\phi \circ \psi\} = A\{\phi\}\{\psi\} \).
- **Tm-Id.** Moreover, for any term \( \sigma \in \text{Tm}(\Gamma, A) \), we have \( \sigma\{\text{id}_\Gamma\} = \sigma \).
- **Tm-Comp.** Under the same assumption, we have \( \sigma\{\phi \circ \psi\} = \sigma\{\phi\}\{\psi\} \).
- **Cons-L.** \( p(A) \circ (\phi, \tau)_A = \phi \).
- **Cons-R.** \( v_A\{\phi, \tau\}_A = \tau \).
- **Cons-Nat.** \( (\phi, \tau)_A \circ \psi = (\phi \circ \psi, \tau\{\psi\})_A \).
- **Cons-Id.** \( (p(A), v_A)_A = \text{id}_{\Gamma.A} \).

We now define our CwF of predicative games and generalized strategies.

Definition 5.5.2 (The CwF \( \mathcal{EPG} \)). We define the CwF \( \mathcal{EPG} \) of predicative games and generalized strategies to be the structure \( \langle \mathcal{EPG}, \text{Ty}, \text{Tm}, \_\{\_\}, \_\{\_\}, \_\{\_\}, \_\{\_\}, \_\{\_\}, \text{Ty-Id.}, \text{Tm-Comp.} \rangle \), where:

- The underlying category \( \mathcal{EPG} \) is the category \( \mathcal{PG} \) of predicative games and generalized strategies defined in Definition 4.3.1.
- For each game \( \Gamma \in \mathcal{EPG} \), we define \( \text{Ty}(\Gamma) \) to be the set of dependent games over \( \Gamma \).
- For a game \( \Gamma \in \mathcal{EPG} \) and a dependent game \( A \in \text{Ty}(\Gamma) \), we define \( \text{Tm}(\Gamma, A) \) to be the set of generalized strategies on the dependent function space \( \prod(\Gamma, A) \).
For each generalized strategy \( \phi : \Delta \to \Gamma \) in \( EPG \), the function \( \mathcal{L}(\phi) : Ty(\Gamma) \to Ty(\Delta) \) is defined by \( A(\phi) \equiv \{ A(\phi \bullet \delta) | \delta : \Delta \} \) for all \( A \in Ty(\Gamma) \), and the functions \( \mathcal{L}(\phi)_A \) are defined by \( \sigma(\phi)_A \equiv \sigma \bullet \phi \) for all \( A \in Ty(\Gamma), \sigma \in Tm(\Gamma, A) \).

- \( I \) is the empty game \( \emptyset, \emptyset, \emptyset, (\{ \} \) \).
- For a game \( \Gamma \in EPG \) and a dependent game \( A \in Ty(\Gamma) \), the comprehension \( \Gamma.A \in EPG \) of \( A \) in \( \Gamma \) is defined to be the dependent pair space \( \sum(\Gamma, A) \).
- The first projections \( p(A) : \sum(\Gamma, A) \to \Gamma \), where \( A \) is a dependent game on \( \Gamma \), are the derelictions \( \text{der}_A \) on \( \Gamma \) up to the “tags” for disjoint union.
- The second projections \( v_A : \prod(\sum(\Gamma, A), A\{p(A)\}) \) are the derelictions \( \text{der}_A \) on \( \sum A \) up to the “tags” for disjoint union.
- For a dependent game \( A \) over a game \( \Gamma \in EPG \) and generalized strategies \( \phi : \Delta \to \Gamma \), \( \tau : \prod(\Delta, A\{\phi\}) \) in \( EPG \), we define \( \langle \phi, \tau \rangle_A \equiv \phi \& \tau : \Delta \to \sum(\Gamma, A) \).

Of course, we need to establish the following:

- **Theorem 5.5.3 (Well-defined \( EPG \)).** The structure \( EPG \) forms a category with families.

**Proof.** First, it is almost straightforward to check that each component is well-defined except for the functions \( \mathcal{L}(\phi)_A \) and the extensions \( \mathcal{L}(\phi)_A \). Let \( A \) be a dependent game over a game \( \Gamma \in EPG \) and \( \phi : \Delta \to \Gamma \), \( \sigma : \prod(\Gamma, A), \tau : \prod(\Delta, A\{\phi\}) \) strategies in \( EPG \).

The composition \( \sigma(\phi)_A \equiv \sigma \bullet \phi \) is a well-defined strategy on the dependent function space \( \prod(\Delta, A\{\phi\}) \) because if \( \delta : \Delta \), then \( \sigma(\phi) \bullet \delta = \sigma \bullet (\phi \bullet \delta) : A(\phi \bullet \delta) = A(\phi)\{\delta\} \). Thus, \( \mathcal{L}(\phi)_A \) is a well-defined function from \( Tm(\Gamma, A) \) to \( Tm(\Delta, A\{\phi\}) \).

For the paring \( \langle \phi, \tau \rangle_A = \phi \& \tau : \Delta \to \Gamma \& (\sum A) \) (it is by Theorem 4.2.2), let \( \delta : \Delta \); we have to show \( \phi \& \tau : \Sigma(\Gamma, A) \). But it is immediate because \( \phi \& \tau : \delta = (\phi \bullet \delta) \& (\tau \bullet \delta) \) and \( \tau \bullet \delta \) is a strategy on the game \( A(\phi)\{\delta\} = A(\phi \bullet \delta) \).

Next, we verify the required equations:

- **Ty-Id.** For any pair of a game \( \Gamma \in EPG \) and a dependent game \( A \) on \( \Gamma \), we have:
  \[
  A\{id_\Gamma\} = \{ A(\text{der}_A \bullet \delta) | \delta : \Delta \} \\
  = \{ A(\delta) | \delta : \Delta \} \\
  = A.
  \]

- **Ty-Comp.** In addition, for any games \( \Delta, \Theta \in EPG \) and strategies \( \psi : \Theta \to \Delta \), \( \phi : \Delta \to \Gamma \) in \( EPG \), we have:
  \[
  A\{\phi \bullet \psi\} = \{ A((\phi \bullet \psi) \bullet \theta) | \theta : \Theta \} \\
  = \{ A(\phi \bullet (\psi \bullet \theta)) | \theta : \Theta \} \\
  = \{ A\{\psi\)(\theta) | \theta : \Theta \} \\
  = A\{\phi\}{\psi}.
  \]
Tm-Id. Moreover, for any strategy $\sigma : \hat{\Pi}(\Gamma, A)$, we have:

$$\sigma \{ \text{id}_\Gamma \} = \sigma \cdot \text{der}_\Gamma = \sigma.$$  

Tm-Comp. Under the same assumption, we have:

$$\sigma \{ \phi \cdot \psi \} = \sigma \cdot (\phi \cdot \psi)$$
$$= (\sigma \cdot \phi) \cdot \psi$$
$$= \sigma \{ \phi \} \cdot \psi$$
$$= \sigma \{ \phi \} \{ \psi \}.$$  

Cons-L. Additionally, for any strategy $\tau : \hat{\Pi}(\Delta, A\{\phi}\})$, we clearly have:

$$p(A) \cdot \langle \phi, \tau \rangle_A = \phi.$$  

Cons-R. Under the same assumption, we have:

$$v_A \{ \langle \phi, \tau \rangle_A \} = v_A \cdot \phi & \tau = \tau.$$  

Cons-Nat.  

$$\langle \phi, \tau \rangle_A \cdot \psi = \phi & \tau \cdot \psi = (\phi \cdot \psi) & (\tau \cdot \psi) = \langle \phi \cdot \psi, \tau \{ \psi \} \rangle_A.$$  

Cons-Id.  

$$\langle p(A), v_A \rangle_A = p(A) & v_A = \text{der}_\sum_{\Gamma, A} = \text{id}_{\Gamma, A}.$$  

5.6 Game-theoretic Type Formers

Note that a CwF handles only the “core” of intuitionistic type theory: It interprets just the syntax common to all the types. Thus, for a “full” interpretation of the type theory, we need to equip EPG with additional structures to interpret $\prod$, $\sum$, and Id-types, as well as universes in the type theory. This is the aim of the present section; we consider each type in order.

5.6.1 Game-theoretic Dependent Function Types

We begin with $\Pi$-types. First, we recall the general, categorical interpretation of $\Pi$-types.

Definition 5.6.1 (CwFs with $\Pi$-types [Hof97]). A CwF $C$ is said to support $\Pi$-types if:

- $\Pi$-Form. For any context $\Gamma \in C$ and types $A \in \text{Ty}(\Gamma)$, $B \in \text{Ty}(\Gamma, A)$, there is a type $\Pi(A, B) \in \text{Ty}(\Gamma)$.

- $\Pi$-Intro. Additionally, if $\sigma \in \text{Tm}(\Gamma, A, B)$, then there is a term $\lambda_{A,B}(\sigma) \in \text{Tm}(\Gamma, \Pi(A, B))$.  

34
We now propose our game-theoretic definitions.

- **Π-Elim.** Under the same assumption, for any terms $\kappa \in \text{Tm}(\Gamma, [\prod(A, B)])$, $\tau \in \text{Tm}(\Gamma, A)$, there is a term
  \[
  \text{App}_{A, B}(\kappa, \tau) \in \text{Tm}(\Gamma, B(\overline{\tau}))
  \]
  where $\overline{\tau} \overset{df}{=} (\text{id}_\Gamma, \tau)_A : \Gamma \to \Gamma.A$.

- **Π-Comp.** We have the equation
  \[
  \text{App}_{A, B}(\lambda_{A, B}(\sigma), \tau) = \sigma(\overline{\tau}).
  \]

- **Π-Subst.** Moreover, for any context $\Delta \in \mathcal{C}$ and morphism $\phi : \Delta \to \Gamma$ in $\mathcal{C}$, we have
  \[
  [\prod(A, B)](\phi) = [\prod(A, B)](A\{\phi \}, B(\phi^+))
  \]
  where $\phi^+ \overset{df}{=} \langle \phi \circ p(A\{\phi \}), v_{A\{\phi \}} \rangle_A : \Delta.A\{\phi \} \to \Gamma.A$.

- **λ-Subst.** Under the same assumption, for any term $\epsilon \in \text{Tm}(\Gamma.A, B)$, we have
  \[
  \lambda(\epsilon\{\phi \}) = \lambda(\epsilon(\phi^+)) \in \text{Tm}(\Delta, [\prod(A, B)](A\{\phi \}, B(\phi^+)))
  \]
  where note that $[\prod(A, B)](A\{\phi \}, B(\phi^+)) \in \text{Ty}(\Delta)$.

- **App-Subst.** Finally, we have
  \[
  \text{App}(\kappa, \tau\{\phi \}) = \text{App}(\kappa\{\phi \}, \tau\{\phi \}) \in \text{Tm}(\Delta, B(\overline{\tau} \circ \phi))
  \]
  where note that $\kappa(\phi) \in \text{Tm}(\Delta, \text{Ty}(\Gamma.A, B(\phi^+)))$, $\tau(\phi) \in \text{Tm}(\Delta, A\{\phi \})$, and $\phi^+ \circ \tau(\phi) = \langle \phi \circ p(A\{\phi \}), v_{A\{\phi \}} \rangle_A \circ (\text{id}_\Delta, \tau\{\phi \})_{A\{\phi \}} = \langle \phi, \tau(\phi) \rangle_A = (\text{id}_\Gamma, \tau)_A \circ \phi = \overline{\tau} \circ \phi$.

We now propose our game-theoretic $\prod$-types.

- **Proposition 5.6.2 (EPΓ supports $\prod$-types).** The CwF EPΓ supports $\prod$-types.

**Proof.** Let $\Gamma \in \text{EPΓ}$, $A \in \text{Ty}(\Gamma)$, $B \in \text{Ty}(\Gamma.A)$, $\sigma \in \text{Tm}(\Gamma.A, B)$ in EPΓ.

- **Π-Form.** We need to generalize the construction of dependent function space, as $A$ itself may be a dependent game. Then we define $[\prod(A, B)] \overset{df}{=} \{[\prod(A, B)]_{\gamma} : \Gamma \in \text{Ty}(\Gamma)\}$, where the dependent game $B_{\gamma} : A\gamma \to \mathcal{U}$ is defined by $B_{\gamma}(\sigma) \overset{df}{=} B(\gamma \& \sigma)$ for all $\gamma : \Gamma, \sigma : A\gamma$. Note that if $\Gamma = \emptyset$, then $[\prod(A, B)] = \{[\prod(A, B)]\}$; thus $\prod$ is a generalization of $\hat{\prod}$, and we call $[\prod(A, B)]$ the dependent function space of $B$ over $A$ as well.

- **Π-Intro.** Because we have the obvious correspondence $\hat{\prod}([\sum(\Gamma, A), B]) \cong \hat{\prod}(\Gamma, [\prod(\Gamma, A, B)])$ up to the “tags” of moves for disjoint union, we may obtain $\lambda_{A, B}(\sigma) : \hat{\prod}(\Gamma, [\prod(\Gamma, A, B)])$ from any $\sigma : \hat{\prod}([\sum(\Gamma, A), B])$ by “adjusting the tags”.

- **Π-Elim.** For $\kappa : \hat{\prod}(\Gamma, [\prod(\Gamma, A, B)])$ and $\tau : \hat{\prod}(\Gamma, A)$, we define $\text{App}_{A, B}(\kappa, \tau) \overset{df}{=} \text{ev} \bullet (\kappa \& \tau)$, where ev is the evaluation strategy (see [A+97] for the precise definition in which it is called application or modus ponens). It is not hard to see that $\text{App}_{A, B}(\kappa, \tau) : \hat{\prod}(\Gamma, B(\overline{\tau}))$, where $\overline{\tau} = \text{der}_{\Gamma \& \tau} : \Gamma \to [\sum(\Gamma, A)]$. 

35
*\(\Pi\text{-Comp.}\) By a simple calculation, we have:
\[
\text{App}_{A,B}(\lambda_{A,B}(\sigma), \tau) = \text{ev} \cdot (\lambda_{A,B}(\sigma) \& \tau) = \sigma \cdot (\text{der} \& \tau) = \sigma [\top].
\]

*\(\Pi\text{-Subst.}\) Moreover, for any \(\Delta \in \mathcal{EPG}\) and \(\phi : \Delta \rightarrow \Gamma\) in \(\mathcal{EPG}\), we have:
\[
\Pi(A,B)\{\phi\} = \Pi(A\{\phi\}, B\{\phi^+\})
\]
where \(\text{ev} \cdot (\phi \bullet (A\{\phi\})) \& V_{A(\phi)} : \sum(\Delta, A\{\phi\}) \rightarrow \sum(\Gamma, A)\) and \(B\{\phi^+\} \in \text{Ty}(\sum(\Delta, A\{\phi\}))\).

Note that the third equation holds because
\[
B\{\phi^+\}_\delta(\psi) = B\{\phi^+\}(\delta \& \psi) = B((\phi \bullet (A\{\phi\})) \& V_{A(\phi)} \bullet (\delta \& \psi)) = B((\phi \bullet \delta) \& \psi) = B\delta(\psi)
\]
for all \(\psi : A(\phi \bullet \delta)\).

*\(\lambda\text{-Subst.}\) For any term \(\iota : \prod(\sum(\Gamma, A), B)\), we clearly have:
\[
\lambda(\iota)\{\phi\} = \lambda(\iota) \bullet \phi = \lambda(\iota \bullet ((\phi \bullet (A\{\phi\})) \& V_{A(\phi)})) = \lambda(\iota\{\phi^+\}).
\]

*\(\text{App-Subst.}\) Finally, we have:
\[
\text{App}(\kappa, \tau)\{\phi\} = (\text{ev} \cdot (\kappa \& \tau)) \bullet \phi = \text{ev} \cdot ((\kappa \& \tau) \bullet \phi) = \text{ev} \cdot ((\kappa \bullet \phi) \& (\tau \bullet \phi)) = \text{ev} \cdot (\kappa\{\phi\} \& \tau\{\phi\}).
\]

\[\blacksquare\]

### 5.6.2 Game-theoretic Dependent Pair Types

Next, we consider \(\Sigma\)-types. Again, we begin with the general definition.

*Definition 5.6.3* (CwFs with \(\Sigma\)-types [Hof97]). A CwF \(C\) is said to support \(\Sigma\)-types if:

*\(\Sigma\text{-Form.}\) For any context \(\Gamma \in C\) and types \(A \in \text{Ty}(\Gamma), B \in \text{Ty}(\Gamma.A)\), there is a type
\[
\Sigma(A, B) \in \text{Ty}(\Gamma).
\]

36
Proof. Additionally, there is a morphism
\[ \text{Pair}_{A,B} : \Gamma.A.B \to \Gamma.\sum(A, B) \]
in \( C \).

- \( \sum\text{-Intro} \). For any type \( P \in \text{Ty}(\Gamma, \sum(A, B)) \) and term \( \psi \in \text{Ty}(\Gamma.A.B, P\{\text{Pair}_{A,B}\}) \), there is a term
\[ R^{\sum}_{A,B,P}(\psi) \in \text{Ty}(\Gamma, \sum(A, B), P). \]

- \( \sum\text{-Elim} \). We have the equation
\[ R^{\sum}_{A,B,P}(\psi) \{\text{Pair}_{A,B}\} = \psi. \]

- \( \sum\text{-Comp} \). Moreover, for any context \( \Delta \in C \) and morphism \( \phi : \Delta \to \Gamma \) in \( C \), we have:
\[ \sum(A, B)\{\phi\} = \sum(A\{\phi\}, B\{\phi^+\}) \]
where \( \phi^+ \overset{\text{df}}{=} \langle \phi \circ p(A\{\phi\}), v_{A\{\phi\}} \rangle : \Delta.A\{\phi\} \to \Gamma.A. \)

- \( \text{Pair-Subst} \). Under the same assumption, we have:
\[ p(\sum(A, B)) \circ \text{Pair}_{A,B} = p(A) \circ p(B) \]
\[ \phi^+ \circ \text{Pair}_{A\{\phi\}, B\{\phi^+\}} = \text{Pair}_{A,B} \circ \phi^+ \]
where \( \phi^+ \overset{\text{df}}{=} \langle \phi \circ p(\sum(A, B)\{\phi\}), v_{\sum(A, B)\{\phi\}} \rangle : \Delta.\sum(A, B)\{\phi\} \to \Gamma.\sum(A, B) \)
and
\[ \phi^{++} \overset{\text{df}}{=} \langle \phi^+ \circ p(B\{\phi^+\}), v_{B\{\phi^+\}} \rangle : \Delta.A\{\phi\}.B\{\phi^+\} \to \Gamma.A.B. \]

- \( R^\sum\text{-Subst} \). Finally, we have:
\[ R^\sum_{A,B,P}(\psi)\{\phi^*\} = R^\sum_{A\{\phi\}, B\{\phi^+\}, P\{\phi^+\}}(\psi\{\phi^{++}\}). \]

Now, we describe our game-theoretic interpretation of \( \sum\text{-types} \):

\begin{itemize}
  \item \( \sum\text{-Form} \). Similar to dependent function space, we generalize the construction of the dependent pair space \( \sum \) by \( \sum(A, B) \overset{\text{def}}{=} \{ \sum(A\gamma, B\gamma) \mid \gamma \in \text{Ty}(\Gamma) \}. \)
  \item \( \sum\text{-Intro} \). Note that we have an obvious correspondence \( \sum(\sum(\Gamma, A), B) \cong \sum(\Gamma, \sum(A, B)) \)
up to the “tags” for disjoint union. We define \( \text{Pair}_{A,B} : \sum(\sum(\Gamma, A), B) \to \sum(\Gamma, \sum(A, B)) \)
as the obvious dereliction up to the correspondence.
  \item \( \sum\text{-Elim} \). By the above correspondence, for any dependent game \( P \in \text{Ty}(\sum(\sum(\Gamma, A), B)) \)
and strategy \( \psi : \prod(\sum(\sum(\Gamma, A), B), P\{\text{Pair}_{A,B}\}) \), we may construct the strategy \( R^\sum_{A,B,P}(\psi) : \prod(\sum(\Gamma, \sum(A, B)), P) \) from \( \psi \) by \( R^\sum_{A,B,P}(\psi) \overset{\text{def}}{=} \psi \circ \text{Pair}_{A,B}^{-1}. \)
\end{itemize}
Definition 5.6.5

Next, we consider identity types. Again, we first review the general, categorical interpretation.

5.6.3 Game-theoretic Identity Types

Id-types (CwFs with identity types [Hof97])

- Pair-Subst.
- Id-Form.

\[ \sum (A, B) \{ \phi \} = \{ \sum (A \gamma, B \gamma) | \gamma : \Gamma \} \{ \phi \} \]

\[ = \{ \sum (A(\phi \bullet \delta), B(\phi \bullet \delta)) | \delta : \Delta \} \]

\[ = \{ \sum (A(\phi(\delta)), B(\phi(\delta))) | \delta : \Delta \} \]

\[ = \sum (A \{ \phi \}, B \{ \phi^+ \}) \]

where \( \phi^+ \) is said to be \( \phi \bullet p(\{ A \{ \phi \} \}) ) \& v_{A(\phi)} : \sum (\Delta, A \{ \phi \}) \to \sum (\Gamma, A) \).

- Pair-Subst.

Under the same assumption, we have \( p(\sum (A, B)) \bullet \text{Pair}_{A, B} = p(A) \bullet p(B) \) by the definition, and we also have:

\[ \phi^+ \bullet \text{Pair}_{A(\phi), B(\phi^+)} \]

\[ = (\phi \bullet p(\sum (A, B) \{ \phi \})) \& v_{A(\phi), B(\phi^+)} \bullet \text{Pair}_{A(\phi), B(\phi^+)} \]

\[ = (\phi \bullet p(\sum (A \phi), B(\phi^+))) \& (\text{Pair}_{A(\phi), B(\phi^+)} \bullet \text{Pair}_{A(\phi), B(\phi^+)} ) \]

\[ = (\phi \bullet p(A \phi)) \bullet p(\sum (A \phi), B(\phi^+)) \& v_{A(\phi), B(\phi^+)} \bullet \text{Pair}_{A(\phi), B(\phi^+)} \]

\[ = (\phi \bullet p(A \phi)) \bullet (\phi^+ \bullet p(B(\phi^+))) \& v_{B(\phi^+)} \bullet \text{Pair}_{A \phi, B(\phi^+)} \]

\[ = \text{Pair}_{A, B} \bullet \phi^{++} \]

where \( \phi^* \) is defined as \( (\phi \bullet p(\sum (A, B) \{ \phi \})) \& v_{A(\phi), B(\phi^+)} \).

- \( R^{\bigwedge} \)-Subst.

Finally, we have:

\[ R^{\bigwedge}_{A, B, P}(\phi^*) = \psi \bullet \text{Pair}_{A, B}^{\bigwedge} \bullet ((\phi \bullet p(\sum (A, B) \{ \phi \})) \& v_{A(\phi), B(\phi^+)}) \]

\[ = (\psi \bullet ((\phi^+ \bullet p(B(\phi^+))) \& v_{B(\phi^+)}) \bullet \text{Pair}_{A(\phi), B(\phi^+)}^{\bigwedge} \]

\[ = R^{\bigwedge}_{A(\phi), B(\phi^+), P(\phi^+)}(\psi \bullet ((\phi^+ \bullet p(B(\phi^+))) \& v_{B(\phi^+)}) ) \]

\[ = R^{\bigwedge}_{A(\phi), B(\phi^+), P(\phi^+)}(\psi(\phi^{++})). \]

5.6.3 Game-theoretic Identity Types

Next, we consider identity types. Again, we first review the general, categorical interpretation.

Definition 5.6.5 (CwFs with identity types [Hof97]). A CwF C is said to support identity types (or Id-types) if:

- Id-Form.

For each context \( \Gamma \in C \) and type \( A \in Ty(\Gamma) \), there is a type

\[ \text{Id}_A \in Ty(\Gamma.A.\alpha^+) \]

where \( \alpha^+ \equiv A(p(A)) \in Ty(\Gamma.A) \).
\[ \text{Proposition 5.6.6 \ (}\Gamma \text{Id-Form.} \text{Id-Comp.} \text{Id-Intro.} \text{Id-Subst.} \text{Id-Elim.} \text{Id-Intro.} \text{R} \text{W} \text{e then equip our CwF EPG.} \text{We then have the equation} \]
\[ \text{and morphism} \phi : \Delta \to \Gamma \text{ in } C, \text{we have} \]
\[ \text{Furthermore, for any context } \Delta \in C \text{ and morphism } \phi : \Delta \to \Gamma \text{ in } C, \text{we have} \]
\[ \text{Also, the following equation holds} \]
\[ \text{Finally, we have} \]
\[ \text{We then equip our CwF EPG with game-theoretic Id-types.} \]

\[ \text{Proposition 5.6.6 \ (} EPG \text{ supports identity types). The CwF EPG supports identity types.} \]

\[ \text{Proof. Let } \Gamma \in EPG, A \in \text{Ty}(\Gamma) \text{ and } B \in \text{Ty}(\Gamma \cdot A) \text{ in } EPG, \text{where } A^+ \overset{df}{=} A\{p(A)\} \in \text{Ty}(\Gamma \cdot A). \]

\[ \text{Id-Form. We define the dependent game } \text{Id}_A \in \text{Ty}(\sum(\sum(\Gamma, A), A^+)) \text{ to be:} \]
\[ \{\text{Id}_A(\sigma_1, \sigma_2) | (\gamma & \sigma_1)& \sigma_2 : \sum(\sum(\Gamma, A), A^+)\}. \]

\[ \text{In other words, we define } \text{Id}_A((\gamma & \sigma_1)& \sigma_2) \overset{df}{=} \text{Id}_A(\sigma_1, \sigma_2). \]

\[ \text{Id-Intro. The morphism} \]
\[ \text{is defined to be the strategy that plays as the dereliction between } \sum(\sum(\Gamma, A), A^+_1), \text{ and } A_1 \text{ and } A^+_1, \text{ or on } \text{Id}_A, \text{ where the subscripts are to distinguish the different copies of } A. \]
\[\text{Id-Elim.}\] For each strategy \(\tau : \prod \left( \sum (\Gamma, A), B \{\text{Refl}_A\} \right)\) in \(\mathcal{E}PG\), we define:

\[R_{A,B}^{\text{Id}}(\tau) \overset{\text{df}}{=} \tau \bullet \text{Refl}_{A}^{-1} : \prod \left( \sum (\sum (\sum (\Gamma, A_1), A_2^+), \text{Id}_A), B \right).\]

\[\text{Id-Comp.}\] We then clearly have:

\[R_{A,B}^{\text{Id}}(\tau \{\text{Refl}_A\}) = R_{A,B}^{\text{Id}}(\tau) \bullet \text{Refl}_A = \tau \bullet \text{Refl}_{A}^{-1} \bullet \text{Refl}_A = \tau.\]

\[\text{Id-Subst.}\] Furthermore, for any game \(\Delta \in \mathcal{P}G\) and strategy \(\phi : \Delta \rightarrow \Gamma\) in \(\mathcal{E}PG\), we have:

\[\text{Id}_A\{\phi^+\} = \{\text{Id}_A(\sigma_1, \sigma_2) | (\gamma \& \sigma_1) \& A \Rightarrow \sum (\sum (\Gamma, A), A^+)\} \{\phi^+\} = \{\text{Id}_A(\phi) \cdot (\phi^+ \& (\delta \& \tau_1) \& \tau_2) | (\delta \& \tau_1) \& \tau_2 : \sum (\sum (\Delta, A \{\phi\}), A \{\phi^+\})\} = \{\text{Id}_A(\phi) \cdot (\tau_1, \tau_2) | (\delta \& \tau_1) \& \tau_2 : \sum (\sum (\Delta, A \{\phi\}), A \{\phi^+\})\} = \text{Id}_A(\phi),\]

where \(\phi^+ \overset{\text{df}}{=} (\phi \cdot p(A \{\phi\}) \& v_{\text{Ad}}(\phi)) \Rightarrow \sum (\sum (\Delta, A \{\phi\}), A \{\phi^+\}) \Rightarrow \sum (\sum (\Gamma, A), A^+).\) Note that

\[\phi^+ \cdot (\delta \& \tau_1) \& \tau_2 = ((\phi^+ \cdot p(A \{\phi^+\})) \& v_{\text{Ad}}(\phi^+)) \cdot (\delta \& \tau_1) \& \tau_2 = (\phi^+ \cdot p(A \{\phi^+\})) \cdot (\delta \& \tau_1) \& \tau_2 \& v_{\text{Ad}}(\phi^+)) \cdot (\delta \& \tau_1) \& \tau_2 = ((\phi \cdot (\delta \& \tau_1) \& \tau_2) = ((\phi \cdot (\delta \& \tau_1) \& \tau_2) \& v_{\text{Ad}}(\phi^+) \cdot (\delta \& \tau_1) \& \tau_2).

\[\text{Refl-Subst.}\] Also, the following equation holds:

\[\text{Refl}_A \cdot \phi^+ = \text{Refl}_A \cdot (\phi \cdot p(A \{\phi\})) \& v_{\text{Ad}}(\phi) = ((\phi^+ \cdot p(\text{Id}_A \{\phi^+\})) \& v_{\text{Id}_A(\phi^+)} \cdot \text{Refl}_A(\phi) = \phi^{++} \cdot \text{Refl}_A(\phi) \overset{\text{df}}{=} (\phi^+ \cdot p(\text{Id}_A \{\phi^+\})) \& v_{\text{Id}_A(\phi^+)}.

\[\text{RId-Subst.}\] Finally, we have:

\[R_{A,B}^{\text{Id}}(\tau) \{\phi^{++}\} = (\tau \bullet \text{Refl}^{-1}_A) \cdot ((\phi^+ \cdot p(\text{Id}_A \{\phi^+\})) \& v_{\text{Id}_A(\phi^+)} = (\tau \cdot ((\phi \cdot p(A \{\phi\})) \& v_{\text{Ad}}(\phi)) \cdot \text{Refl}_A^{-1}(\phi) = R_{A,B}^{\text{Id}}(\tau \cdot (\phi^+ \cdot p(A \{\phi\})) \& v_{\text{Ad}}(\phi)) = R_{A,B}^{\text{Id}}(\tau \cdot (\phi^+) = R_{A,B}^{\text{Id}}(\phi \cdot (\delta \& \tau_1) \& \tau_2) = (\phi \cdot (\delta \& \tau_1) \& \tau_2) \& v_{\text{Ad}}(\phi^+)) \cdot (\delta \& \tau_1) \& \tau_2.

Note that equality of strategies \(\tau_1, \tau_2 : G \text{ "on the nose" is equivalent to an inhabitation of the Id-game Id_G(\tau_1, \tau_2) by a non-trivial strategy. Thus, EP\!G induces a model of }\textit{extensional }\textit{variant of the type theory [Hof97].}
5.6.4 Game-theoretic Universes

In a completely analogous way, we now define a general, categorical notion of CwFs with a hierarchy of universes, following the formulation in [Uni13].

Definition 5.6.7 (CwFs with universes). A CwF $\mathcal{C}$ is said to support (a hierarchy of) universes if:

- **U-Form.** For any context $\Gamma \in \mathcal{C}$, there is a type $U_n \in \text{Ty}(\Gamma)$ for each natural number $n \in \mathbb{N}$. We often write $U$ for $U_n$ with some $n \in \mathbb{N}$.

- **U-Intro.** For any type $A \in \text{Ty}(\Gamma)$ and natural number $n \in \mathbb{N}$, we have:
  
  $$A \in \text{Tm}(\Gamma, U)$$
  
  $$U_n \in \text{Tm}(\Gamma, U_{n+1})$$

  where $U$ refers to $U_k$ for some $k \in \mathbb{N}$.

- **U-Cumul.** For any $n \in \mathbb{N}$, if $A \in \text{Tm}(\Gamma, U_n)$, then $A \in \text{Tm}(\Gamma, U_{n+1})$.

Finally, we present our game-theoretic interpretation of the hierarchy of universes:

Proposition 5.6.8 ($\mathcal{EPG}$ supports universes). The CwF $\mathcal{EPG}$ supports a hierarchy of universes.

Proof. Let $\Gamma \in \mathcal{EPG}$ be any game.

- **U-Form.** The dependent game $U_n \in \text{Ty}(\Gamma)$ is defined to be the “constant” dependent game at the $n^{th}$ universe game $U_n$ for all $n \in \mathbb{N}$.

- **U-Intro.** Any dependent game $G \in \text{Ty}(\Gamma)$ is clearly a strategy $G : \prod(\Gamma, U)$ by the definition. Similarly, the strategy $U_n : U_{n+1}$ for each $n \in \mathbb{N}$ induces the “constant” strategy $U_n : \prod(\Gamma, U_{n+1})$.

- **U-Cumul.** If $G : \prod(\Gamma, U_n)$, then clearly $G : \prod(\Gamma, U_{n+1})$ by the definition of the universe games.

\[\text{20\ Here, we abuse the notation, but it will not cause any confusion in practice.}\]
6  Game-theoretic Interpretation of ITT

Recall that the model in $EPG$ is an interpretation of the extensional variant of the type theory (ETT). To construct a model of the intensional variant (ITT), we need to consider different interpretation of identity types. A reasonable idea is to relax the existence of a copy-cat strategy (as a proof of equality) to the existence of an isomorphism strategy 21.

6.1 Isomorphism Strategies as Proofs of Equalities

As we shall see, to interpret proofs of equalities by isomorphism strategies, we need to focus on strategies that preserve isomorphisms between strategies:

Definition 6.1.1 (Iso-preserving strategies). A strategy $\tau : A \to B$ is said to be iso-preserving if there is an isomorphism strategy $\tau_p : \tau \circ \sigma \to \tau \circ \sigma'$ for any $\sigma, \sigma' : A, p : \sigma \to \sigma'$.

In particular, a dependent game $B : A \to U$ is iso-preserving iff $\sigma \equiv \sigma' : A \Rightarrow B\sigma \equiv B\sigma' : U$ for all $\sigma, \sigma' : A$. That is, the elements of an iso-preserving dependent game are not completely arbitrary but weakly unified via isomorphisms. We need this condition, since otherwise, e.g., Leibniz' law would not hold in our game semantics.

We proceed to show that isomorphisms are preserved under some operations:

Lemma 6.1.2 (Preservation of isomorphisms). Isomorphisms between strategies are preserved under tensor product, paring, and promotion, and isomorphisms between games are preserved under tensor product, linear implication, product, and exponential.

Proof. We only handle the case of games; the case of strategies are completely analogous. Let $\tau : A \to B, \kappa : C \to D$ be isomorphism strategies. Then we have the following isomorphisms:

$$
\tau \otimes \kappa : A \otimes C \equiv B \otimes D \quad (\text{with } (\tau \otimes \kappa)^{-1} = \tau^{-1} \otimes \kappa^{-1})
$$

$$
\tau \& \kappa : A \& C \equiv B \& D \quad (\text{with } (\tau \& \kappa)^{-1} = \tau^{-1} \& \kappa^{-1})
$$

$$
!\tau : !A \equiv !B \quad (\text{with } (!\tau)^{-1} = !(\tau^{-1}))
$$

$$
\tau^{-1} \circ \kappa : (A \circ C) \equiv (B \circ D) \quad (\text{with } (\tau^{-1} \circ \kappa)^{-1} = \tau^{-1} \circ \kappa^{-1}).
$$

Note that isomorphisms are not preserved under composition. For instance, consider the constant zero strategy $z : \prod (N, \text{Fin}^N)$ introduced in the previous example. For any natural numbers $m, n \in N$ with $m \neq n$, we have $m \equiv n$ but $z \cdot m \not\equiv z \cdot n$. This illustrates why we have to focus on iso-preserving strategies.

6.2 Game-theoretic Category with Families for ITT

We now define another CwF of games and strategies that interprets ITT.

Definition 6.2.1 (The intensional CwF $IPG$). The CwF $IPG$ is defined to be almost the same as $EPG$ except that the strategies are restricted to iso-preserving ones. 21

However, note that we have $\equiv m : N$ for all $n, m \in N$ (even if $n \neq m$); thus, if we take isomorphism strategies as proofs of equalities, then we need to consider a different natural numbers game.
It is then immediate to see that:

- Proposition 6.2.2 (Well-defined $\mathcal{IPG}$). The structure $\mathcal{IPG}$ forms a well-defined CwF.

Proof. First, we show that the function

$$\zeta(\phi) : \text{Ty}(\Gamma) \to \text{Ty}(\Delta)$$

$$\{ A \gamma \mid \gamma : \Gamma \} \mapsto \{ A(\phi \bullet \delta) \mid \delta : \Delta \}$$

for any strategy $\phi : \Delta \to \Gamma$ in $\mathcal{IPG}$ is well-defined. But it is immediate because if $\delta \equiv \delta' : \Delta$, then $\phi \bullet \delta \equiv \phi \bullet \delta'$ as $\phi$ is iso-preserving so that $A(\phi \bullet \delta) \equiv A(\phi \bullet \delta')$ as $A$ is iso-preserving, showing that $A(\phi)$ is an iso-preserving dependent game.

Next, note that the function

$$\zeta(\phi)_A : \hat{\Pi}(\Gamma, A) \to \hat{\Pi}(\Delta, A(\phi))$$

$$\sigma \mapsto \sigma \bullet \phi$$

for each strategy $\phi : \Delta \to \Gamma$ in $\mathcal{IPG}$ is well-defined because $\sigma$ and $\phi$ are both iso-preserving.

Also, derelictions are clearly iso-preserving, so the first and second projections are well-defined. Finally, by Lemma 6.1.2, the extensions are well-defined, completing the proof. ■

6.3 Game-theoretic Type Formers

We proceed to equip the CwF $\mathcal{IPG}$ with semantic type formers. First, for $\Pi$- and $\Sigma$-types, the same constructions as $\mathcal{EPG}$ can be straightforwardly applied:

- Proposition 6.3.1 ($\mathcal{IPG}$ supports $\Pi$- and $\Sigma$-types). The CwF $\mathcal{IPG}$ supports $\Pi$- and $\Sigma$-types.

Proof. We apply the same constructions as the case of $\mathcal{EPG}$. First, the constructions of $\Pi$ and $\Sigma$ are well-defined because the iso-preserving property of dependent games is preserved by Lemma 6.1.2.

Next, we have no problem in inheriting the interpretation of $\Pi$-Intro and $\Pi$-Elim rules of $\mathcal{EPG}$ because $\lambda_{A,B}(\sigma)$ is essentially $\sigma$, and the strategy $\text{ev}$ is essentially a dereliction.

Also, $\Sigma$-Intro and $\Sigma$-Elim rules can be interpreted in the same way as $\mathcal{EPG}$ because $\text{Pair}_{A,B}$ is essentially a dereliction, and $R_{A,B,P}(\psi) = \psi \bullet \text{Pair}_{A,B}$. Finally, the equations hold just in the same way as the case of $\mathcal{EPG}$, completing the proof. ■

Next, we consider intensional Id-types, in which the iso-preserving property of dependent games plays an important role.

- Notation. Let $A, B$ be predicative games, and $\sigma : A, \tau : B$ strategies. If $\phi(\sigma_1, \ldots, \sigma_n) : B$ is a composition of strategies $\sigma_1 : A_1, \ldots, \sigma_n : A_n$ and possibly some other strategies, then $(\sigma_1 \otimes \cdots \otimes \sigma_n) = \phi(\sigma_1, \ldots, \sigma_n)$ denotes the strategy on $A \to B$, where $A \df A_1 \otimes \cdots \otimes A_n$ that plays as $\phi(\sigma_1, \ldots, \sigma_n)$ plus “copy-cat” between the two occurrences of $\sigma_i$ for all $i = 1, \ldots, n$.

Abusing the notation, we apply these notations for the implication $A \to B = !A \to B$ as well.

- Proposition 6.3.2 ($\mathcal{IPG}$ supports intensional Id-types). The CwF $\mathcal{IPG}$ supports (intensional) identity types.

Proof. Let $\Gamma \in \mathcal{IPG}$, $A \in \text{Ty}(\Gamma)$ and $B \in \text{Ty}(\Gamma.A.A^+.\text{Id}_A)$ in $\mathcal{IPG}$, where $A^+ \df A\{p(A)\} \in \text{Ty}(\Gamma.A)$.

\footnote{Strictly speaking, the compositions in $\phi(\sigma_1, \ldots, \sigma_n)$ should be considered as “non-hiding” [YAl0].}
\textbf{Id-Form.} For each triple of a game $G$ and strategies $\tau_1, \tau_2 : G$, we define the \textit{intensional identity game} for $\tau_1$ and $\tau_2$ to be the subgame $\text{Id}_G(\tau_1, \tau_2) \leq \tau_1 \rightarrow \tau_2$ whose strategies are isomorphisms $\tau_1 \cong \tau_2$ only. We then define the dependent game $\text{Id}_A \in \text{Ty}(\sum(\Gamma, A, A^+))$ to be $\{\text{Id}_A(\sigma_1, \sigma_2) : (\gamma & \sigma_1) & \sigma_2 : \sum(\Gamma, A, A^+)\}$. In other words, we define $\text{Id}_A((\gamma & \sigma_1) & \sigma_2) \equiv \text{Id}_A(\sigma_1, \sigma_2)$.

\textbf{Id-Intro.} The morphism

$$\text{Refl}_A : \sum(\Gamma, A_1) \rightarrow \sum(\sum(\Gamma, A_2), A_1^+), \text{Id}_A)$$

is defined to be the strategy that plays as the dereliction between $\sum(\Gamma, A_1)$ and $\sum(\Gamma, A_2)$, $A_1$ and $A_1^+$, or on $\text{Id}_A$, where the subscripts are to distinguish the different copies of $A$.

\textbf{Id-Elim.} For each strategy $\tau : \prod(\sum(\Gamma, A), B(\text{Refl}_A))$, we need to define a strategy

$$\text{R}_A^\text{ld}(A, B)(\tau) : \prod(\prod(\sum(\sum(\Gamma, A_1), A_1^+), \text{Id}_A), B).$$

Note that we cannot simply apply the same construction as in the case of $\mathcal{E}PG$ because the 3rd and 4th elements in $\sum(\sum(\sum(\sum(\Gamma, A_2), A_1^+), \text{Id}_A))$ may be “replaced” by the strategy $\text{Refl}_A \bullet \text{Refl}_A^{-1}$. Observe that $\text{Refl}_A \bullet (\gamma \& \sigma_1) \equiv ((\tau \& \sigma_1) \& \sigma_2) \& \alpha : \sum(\sum(\sum(\sum(\Gamma, A_2), A_1^+), \text{Id}_A))$ for any $\gamma : \Gamma, \sigma_1, \sigma_2 : A_1$ and $\alpha : \text{Id}_A(\sigma_1, \sigma_2)$ because $\alpha$ is an isomorphism. Then, since $B$ is iso-preserving, we have $B(\text{Refl}_A \bullet (\gamma \& \sigma_1)) \equiv B((\tau \& \sigma_1) \& \sigma_2) : \mathcal{U}$. Writing $B_{\gamma, \sigma_1, \sigma_2, \alpha}^\text{ld}$ for the isomorphism, we may define $\text{R}_A^\text{ld}(A, B)(\tau)$ to be:

&\equiv ((\gamma \& \sigma_1) \& \sigma_2) \& \alpha \equiv B_{\gamma, \sigma_1, \sigma_2, \alpha}^\text{ld} \bullet (\gamma \& \sigma_1) \equiv B_{\gamma, \sigma_1, \sigma_2, \alpha}^\text{ld} \bullet \tau \bullet (\gamma \& \sigma_1) \equiv \sum(\sum(\sum(\sum(\Gamma, A_2), A_1^+), \text{Id}_A))

where we take a dereliction as $B_{\gamma, \sigma_1, \sigma_2, \alpha}^\text{ld}$ when $\alpha = \sigma_2$ and $\alpha = \text{der}_\sigma$.

\textbf{Id-Comp.} We then clearly have $\text{R}_A^\text{ld}(A, B)(\tau) \{\text{Refl}_A\} = \text{R}_A^\text{ld}(A, B)(\tau) \bullet \text{Refl}_A = \tau$, where note that $B_{\gamma, \sigma_1, \sigma_2, \alpha}^\text{ld}$ in this case is a dereliction.

The remaining equations for substitutions hold in the same way as the interpretation of Id-types in $\mathcal{E}PG$, except for the following:

\textbf{Rd-Subst.} Finally, we have:

$$\text{R}_A^\text{ld}(A, B)(\tau)(\phi^{++})$$

\[
= (\text{R}_A^\text{ld}(A, B)(\tau) \bullet (\phi^{++} \bullet p(\text{Id}_A(\phi^{++})))) &\vee \text{Id}_A(\phi^{++}) \\
= &\equiv ((\delta \& \sigma_1) \& \sigma_2) \& \alpha \equiv B_{\phi, \delta, \sigma_1, \sigma_2, \alpha}^\text{ld} \bullet (\phi \bullet (\delta \& \sigma_1) \& \sigma_2) \& \alpha \equiv \sum(\sum(\Delta, A(\phi), A^+ \{\phi^{++}\})), \text{Id}_A(\phi)) \\
= &\equiv B_{\phi, \phi^{++}}(\tau \bullet (\phi \bullet p(A(\phi)) \& \text{Id}_A(\phi))) \\
= &\equiv B_{\phi, \phi^{++}}(\tau(\phi^{++})) \\
\]

which completes the proof.

Finally, we equip $\mathcal{TGP}$ with the game-theoretic universes.

\textbf{Proposition 6.3.3 ($\mathcal{TGP}$ supports universes).} \textit{The CwF $\mathcal{TGP}$ supports a hierarchy of universes.}

\textit{Proof.} It is obvious that we can equip $\mathcal{TGP}$ with the same universes as those of $\mathcal{E}PG$. 

\[44\]
7 Intensionality

We now investigate how intensional the model of ITT in \( \mathcal{IG} \) is through some of the rules in the type theory. For a type \( A \) and terms \( a, a' : A \), we write \( a =_A a' \) or just \( a = a' \) for the Id-type between \( a, a' \) and \( \vdash a \equiv a' : A \) for the judgemental equality between \( a, a' \) in ITT. Also for brevity, we do not notationally distinguish syntactic objects and their interpretations in \( \mathcal{IG} \), i.e., we omit the semantic bracket \( [\cdot] \).

7.1 Equality Reflection

The principle of equality reflection (EqRefl), which states that if two terms are propositionally equal, then they are judgementally equal too, is the difference between ITT and ETT: Roughly, ETT is “ITT plus EqRefl”.

Note that the model in \( \mathcal{IG} \) refutes EqRefl, as two isomorphic strategies are not necessarily equal (e.g., consider strategies \( m, n : N \) with \( m \neq n \)). Hence, it is a model of ITT, not ETT.

7.2 Function Extensionality

Next, we consider the axiom of function extensionality (FunExt), which states that for any type \( A \), dependent type \( B : A \rightarrow U \), and terms \( f, g : \prod_{x : A} B(x) \), we can inhabit the type

\[
\prod_{x : A} f(x) = g(x) \rightarrow f = g.
\]

It is not hard to see that the model in \( \mathcal{IG} \) refutes this axiom: By the definition, a strategy \( \phi = &\{\phi_\sigma | \sigma : A \} : \prod(A, B) \) is not completely specified by the extensional collapse \( \pi_\phi : \text{gs}(A) \rightarrow \text{gs}(\sqcup B) \) and a strategy on \( \sqcup B \); thus, even if \( \prod_{x : A} f(x) = g(x) \) holds, i.e., \( f(x) \) and \( g(x) \) are interpreted as isomorphic strategies on \( \sqcup B \) for all \( x : A \), the interpretation of \( f \) may not be isomorphic to that of \( g \) (e.g., consider the strategies \( z, \hat{z} : \prod(N, \text{Fin}^N) \) where \( z \) is the “constant zero” strategy in the previous example and \( \hat{z} \) is its trivial modification whose maximal play is \( q_1 q_2 0 q_3 0 \ldots q_k 0 \)).

7.3 Uniqueness of Identity Proofs

Next, we investigate the principle of uniqueness of identity proofs (UIP), which states that for any type \( A \), the following type can be inhabited:

\[
\prod_{a_1, a_2 : A} \prod_{p, q : a_1 = a_2} p = q.
\]

It is sound for the model in \( \mathcal{IG} \) by the definition of the Id-types: For any game \( A \) and strategies \( \sigma_1, \sigma_2 : A \), any strategies \( p, q : \sigma_1 = \sigma_2 \) are isomorphism strategies, so there must be an isomorphism strategy between them.

7.4 Criteria of Intensionality

There are Streicher’s three Criteria of Intensionality [Str93]:

- I. \( A : U, x, y : A, z : x =_A y \ \vdash \ x \equiv y : A \).
- II. \( A : U, B : A \rightarrow U, x, y : A, z : x =_A y \ \vdash \ B(x) \equiv B(y) : U \).
- III. If \( \vdash p : t = A \ t' \), then \( \vdash t \equiv t' : A \).

It is straightforward to see that the model in \( \mathcal{IG} \) validates the criteria I and II but refutes the criterion III.
7.5 Univalence

We finally analyze the univalence axiom (UA), the heart of HoTT, which states that

\[(A =_U B) \simeq (A \simeq B)\]

for all types \(A\) and \(B\). Roughly, a term of an equivalence \(A \simeq B\) consists of functions \(f : A \to B,\ g : B \to A\) and propositional equalities \(p : f \circ g = \text{id}_B,\ q : g \circ f = \text{id}_A\); for the precise definition, see [Uni13]. It is then not hard to see that the model in \(\mathcal{IPG}\) does not validate this axiom, as \((A \simeq B)\) carries more components than \((A =_U B)\).

As we have just observed, the interpretation of Id-types in \(\mathcal{IPG}\) does not capture well the properties of Id-types in HoTT.

8 Conclusion

In the present paper, we have presented a new variant of games and strategies that interprets both extensional and intensional variants of intuitionistic type theory with \(\Pi, \Sigma,\) and Id-types as well as universes (with Example 4.1.7, it is not hard to interpret natural numbers type as well).

It generalizes the existing notion of games and strategies and achieves a computational and intensional interpretation of dependent types in an intuitive manner. Remarkably, it interprets the hierarchy of universes as games (perhaps for the first time in the literature).

However, its interpretation of intensional Id-types does not capture phenomena in HoTT. Meanwhile, we have recognized that \(\mathcal{IPG}\) looks quite similar to the CwF of groupoids in the historic paper [HS98]. Thus, it is a future work to refine the model by equipping it with a groupoid structure to interpret Id-types better. Also, it remains to establish a definability result; for this, we need to focus on total and effective games and strategies. Finally, it would be fruitful to incorporate the “dynamic structure” introduced in [YA16] because the distinction between canonical and non-canonical elements [ML84] in intuitionistic type theory may be captured by the notion of explicit and non-explicit strategies in a mathematically precise way.

Acknowledgements

The author was supported by Funai Overseas Scholarship. Also, he is grateful to Samson Abramsky, Kohei Kishida, and Sam Staton for fruitful discussions.
References

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

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

[AJ94] Samson Abramsky and Radha Jagadeesan. Games and full completeness for multiplicative linear logic. *The Journal of Symbolic Logic*, 59(02):543–574, 1994.

[AJM00] Samson Abramsky, Radha Jagadeesan, and Pasquale Malacaria. Full abstraction for PCF. *Information and Computation*, 163(2):409–470, 2000.

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

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

[AM98] Samson Abramsky and Guy McCusker. Call-by-value games. In *Computer Science Logic*, pages 1–17. Springer, 1998.

[BCH14] Marc Bezem, Thierry Coquand, and Simon Huber. A model of type theory in cubical sets. In *19th International Conference on Types for Proofs and Programs (TYPES 2013)*, volume 26, pages 107–128, 2014.

[CAB+86] RL Constable, SF Allen, HM Bromley, WR Cleaveland, JF Cremer, RW Harper, DJ Howe, TB Knoblock, NP Mendler, P Panangaden, et al. Implementing mathematics with the nuprl proof development system. 1986.

[Car86] John Cartmell. Generalized algebraic theories and contextual categories. *Annals of Pure and Applied Logic*, 32(3):209–243, 1986.

[CH88] Thierry Coquand and Gerard Huet. The calculus of constructions. *Information and computation*, 76(2):95–120, 1988.

[Coq98] Catarina Coquand. A realizability interpretation of martin-löf’s type theory. *Twenty-Five Years of Constructive Type Theory*, 1998.

[Dyb96] Peter Dybjer. Internal type theory. In *Types for Proofs and Programs*, pages 120–134. Springer, 1996.

[HO00] J Martin E Hyland and C-HL Ong. On full abstraction for PCF: I, II, and III. *Information and computation*, 163(2):285–408, 2000.

[Hof97] Martin Hofmann. Syntax and semantics of dependent types. In *Extensional Constructs in Intensional Type Theory*, pages 13–54. Springer, 1997.

[HS98] Martin Hofmann and Thomas Streicher. The groupoid interpretation of type theory. *Twenty-five years of constructive type theory (Venice, 1995)*, 36:83–111, 1998.
[HY97] Kohei Honda and Nobuko Yoshida. Game theoretic analysis of call-by-value computation. In *Automata, Languages and Programming*, pages 225–236. Springer, 1997.

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

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

[ML82] Per Martin-Löf. Constructive mathematics and computer programming. *Studies in Logic and the Foundations of Mathematics*, 104:153–175, 1982.

[ML84] Per Martin-Löf. *Intuitionistic Type Theory: Notes by Giovanni Sambin of a series of lectures given in Padova, June 1980*. 1984.

[ML98] Per Martin-Löf. An intuitionistic theory of types. *Twenty-five years of constructive type theory*, 36:127–172, 1998.

[MT11] Andrzej S Murawski and Nikos Tzevelekos. Game semantics for good general references. In *Logic in Computer Science (LICS), 2011 26th Annual IEEE Symposium on*, pages 75–84. IEEE, 2011.

[Nic94] Hanno Nickau. Hereditarily sequential functionals. In *Logical Foundations of Computer Science*, pages 253–264. Springer, 1994.

[Nor07] Ulf Norell. *Towards a practical programming language based on dependent type theory*, volume 32. Citeseer, 2007.

[NPS90] Bengt Nordström, Kent Petersson, and Jan M Smith. Programming in martin-löfs type theory, volume 7 of international series of monographs on computer science, 1990.

[Reu99] Bernhard Reus. Realizability models for type theories. *Electronic Notes in Theoretical Computer Science*, 23(1):128–158, 1999.

[RS84] HE Rose and JC Shepherdson. An intuitionistic theory of types: Predicative part. 1984.

[See84] Robert AG Seely. Locally cartesian closed categories and type theory. In *Mathematical proceedings of the Cambridge philosophical society*, volume 95, pages 33–48. Cambridge Univ Press, 1984.

[Str93] Thomas Streicher. *Investigations into intensional type theory*. 1993.

[T+12] COQ DEVELOPMENT TEAM et al. The coq proof assistant reference manual. *Typical Project (formerly LogiCal)*, 2012.

[Uni13] The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. [https://homotopytypetheory.org/book](https://homotopytypetheory.org/book), Institute for Advanced Study, 2013.

[YA16] Norihiro Yamada and Samson Abramsky. Dynamic games and strategies. *arXiv preprint arXiv:1601.04147*, 2016.