Cocon: Computation in Contextual Type Theory

BRIGITTE PIENTKA, McGill University, Canada
ANDREAS ABEL, Gothenburg University, Sweden
FRANCISCO FERREIRA, Imperial College London, United Kingdom
DAVID THIBODEAU, McGill University, Canada
REBECCA ZUCCHINI, ENS Paris Saclay, France

We describe a Martin-Löf-style dependent type theory, called Cocon, that allows us to mix the intensional function space that is used to represent higher-order abstract syntax (HOAS) trees with the extensional function space that describes (recursive) computations. We mediate between HOAS representations and computations using contextual modal types. Our type theory also supports an infinite hierarchy of universes and hence supports type-level computation—thereby providing metaprogramming and (small-scale) reflection. Our main contribution is the development of a Kripke-style model for Cocon that allows us to prove normalization. From the normalization proof, we derive subject reduction and consistency. Our work lays the foundation to incorporate the methodology of logical frameworks into systems such as Agda and bridges the longstanding gap between these two worlds.

Additional Key Words and Phrases: Dependent Types, Logical Relations, Proof assistants

1 INTRODUCTION

Higher-order abstract syntax (HOAS) is an elegant and deceptively simple idea of encoding syntax and more generally formal systems given via axioms and inference rules. The basic idea is to map uniformly binding structures in our object language to the function space in a meta-language thereby inheriting α-renaming and capture-avoiding substitution. In the logical framework LF [Harper et al. 1993], for example, we encode a simple object language consisting of functions, function application, and let-expressions using a type `tm` together with the constants as follows:

```
lam : (tm → tm) → tm.
app : tm → tm → tm.
letv: tm → (tm → tm) → tm.
```

The object language term $(\mathsf{lam}\ x.\mathsf{lam}\ y.\mathsf{let}\ w = x\ y\ \mathsf{in}\ w\ y)$ is then encoded as `(lam λx.lam λy.letv (app x y) λw. app w y)` using the LF abstractions to model binding. Object level substitution is modelled through LF application; for instance, the fact that $((\mathsf{lam}\ x.M)\ N)$ reduces to $[N/x]M$ in our object language is expressed as `(app (lam M) N)` reducing to `(M N)`. This approach can offer substantial benefits: programmers do not need to build up the basic mathematical infrastructure, they can work at a higher-level of abstraction, encodings are more compact, and hence it is easier to mechanize formal systems together with their meta-theory.

However, this approach relies on the fact that we use an *intensional* function space that lacks recursion, case analysis, inductive types, and universes to adequately represent syntax. In the logical framework LF [Harper et al. 1993] for example we use the dependently-typed lambda calculus as a meta-language to represent formal systems. Two LF objects are equal if they have the same $\beta\eta$-normal form. Under this view, intensional functions represent syntactic binding structures. However, we cannot write recursive programs about such syntactic structures *within* LF, as we lack the power of recursion. We only have a way to represent data. In contrast, to describe computation we rely on the *extensional* function space. Under this view, two functions are (extensionally) equal if they *behave* in the same way, i.e. when they produce equal results when applied to equal inputs. Under this view, functions are opaque.

1.1 Intensional and Extensional Functions – A World of a Difference

To understand the fundamental difference between defining HOAS trees in LF vs. defining HOAS-style trees using inductive types, let us consider an inductive type `D` with one constructor `lam: (D → D) → D`. What is the problem with such a definition in type theory? – In functional ML-like languages, this is, of course, possible, and types like `D` can be explained using domain theory [Scott 1976]. However, the function argument to the constructor `lam` is opaque and we would not be able to pattern match deeper on the argument to inspect the shape and structure of the syntax tree

Authors' addresses: Brigitte Pientka, School of Computer Science, McGill University, Canada, bpientka@cs.mcgill.ca; Andreas Abel, Department of Computer Science and Engineering, Gothenburg University, Sweden, andreas.abel@gu.se; Francisco Ferreira, School of Computer Science, Imperial College London, United Kingdom, f.ferreira-ruiz@imperial.ac.uk; David Thibodeau, School of Computer Science, McGill University, Canada, david.thibodeau@mail.mcgill.ca; Rebecca Zucchini, ENS Paris Saclay, sss, xxx, xx, xxxxx, France, XXX.

that is described by it. We can only observe it by applying it to some argument. The resulting encoding also would not be adequate, i.e. there are terms of type D that are in normal form but do not uniquely correspond to a term in the object language we try to model. As a consequence, we may need to rule out such "exotic" representations [Despeyroux et al. 1995]. But there is a more fundamental problem. In proof assistants based on type theory such as Coq or Agda, we cannot afford to work within an inconsistent system and we demand that all programs we write are terminating. The definition of a constructor `lam` as given previously would be forbidden as it violates what is known as the positivity restriction. Were we to allow it, we can easily write non-terminating programs by pattern matching – even without making a recursive call.

```
apply : D → (D → D)           omega : D                          Omega : D
apply  (lam f) = f            omega = lam (λx → apply x x)       Omega = apply omega omega
```

Here we simply write two functions: the function `apply` unpacks an object of type `D` using pattern matching and the function `omega` creates an object of type `D`. Using `apply` and `omega` we can now write a non-terminating program that will continue to reproduce itself.

This example begs two questions: How can we reason inductively about LF definitions if they are seemingly not inductive? Do we have to simply give up on HOAS definitions to model syntactic structures within type theory to remain consistent?

1.2 Towards Bridging the Gap between Intensional and Extensional Functions

Over the past two decades, we have made substantial progress in bringing the intensional and extensional views closer together. Despeyroux et al. [1997] made the key observation that we can mediate between the weak LF and the strong computation-level function space using a box-modality. The authors describe a simply typed lambda-calculus with iteration and case constructs which preserves the adequacy of higher-order abstract syntax encodings. The well-known paradoxes are avoided through the use of a modal box operator which obeys the laws of S4. In addition to being simply typed, all computation had to be on closed HOAS trees. Despeyroux and Leleu [1999] sketch an extension of this line of work to dependent type theory – however it lacks a normalization proof.

Beluga [Pientka and Cave 2015; Pientka and Dunfield 2010] took another important step towards writing inductive proofs about HOAS trees by generalizing the box-modality to a contextual modal type [Nanevski et al. 2008; Pientka 2008]. This allows us to characterize HOAS trees that depend on a context of assumptions. More importantly, Beluga allows programmers to analyze these contextual HOAS trees using case distinction and recursion. Exploiting the Curry-Howard isomorphism, inductive proofs about HOAS trees can be described using recursive functions. However, the gap between full dependent type theories with recursion and universes such as Martin-Löf type theory, and weak dependent type theories such as LF remains. In particular, Beluga cleanly separates representing syntax from reasoning about syntax. The resulting language is an indexed type system in the tradition of Zenger [1997] and Xi and Pfenning [1999] where the index language is completely different from the computation language which is used to write recursive programs. In Beluga, contextual LF is taken as the index domain. This has the key advantage that meta-theoretic proofs are modular and hinge on the fact that equality in the index domain is decidable. However, this approach also gives up a lot of expressivity; in particular we can only express properties of HOAS trees, but we lack the power to express properties of the functions we write about them. This prevents us from fully exploiting the power of metaprogramming and reflection.

1.3 The Best of Both Worlds

In this paper, we present the Martin-Löf style dependent type theory Cocon where we mediate between intensional syntactic structures and extensional computations using contextual types [Nanevski et al. 2008; Pientka 2008]. Following Beluga, we pair a LF object together with its surrounding LF context and embed it as a contextual object into computations using the box-modality. For example, $\lceil x, y \vdash \mathsf{app}\ x\ y \rceil$ describes a contextual LF object that has the contextual LF type $\lceil x{:}\mathsf{tm}, y{:}\mathsf{tm} \vdash \mathsf{tm} \rceil$. In contrast to Beluga, we also allow computations to be embedded within LF objects. For example, if a program t promises to compute a value $\lceil x{:}\mathsf{tm}, y{:}\mathsf{tm} \vdash \mathsf{tm} \rceil$, then we can embed t directly into an LF object writing $\mathtt{lam}\ \lambda x.\mathtt{lam}\ \lambda y.\mathsf{app}\ \lfloor t \rfloor\ x$. In general, we can use a computation that produces a value of type $\lceil \Psi \vdash A \rceil$ when constructing a LF object in a LF context Φ by unboxing it together with a LF substitution that moves the value from the LF context Ψ to the current LF context Φ. This is written as $\lfloor t \rfloor_\sigma$. In the example, we omitted the substitution as the computation already promised to produce a value in the appropriate LF context.

Being able to embed functions into LF objects is key to express properties about function we write about them. For example, we might implement a function that evaluates a `tm`-object and another function `trans` that eliminates let-expressions from our `tm` language. Then we would like to know whether both the original term and the translated term evaluate to the same value.

Allowing computation within LF objects, might seem like a small change, but it has far reaching consequences. To establish consistency of the type theory, we cannot consider normalization of LF separately from normalization of computations anymore, as it is done in Pientka and Abel [2015] and Jacob-Rao et al. [2018]. As Martin-Löf type theory [1973], Cocon is a predicative type theory and supports an infinite hierarchy of universes. This allows us to write type-level computation, i.e. we can compute types whose shape depends on a given value. Such recursively defined types are sometimes called large eliminations [Werner 1992]. Due to the presence of type-level computations dependencies cannot be erased from the model. As a consequence, the simpler proof technique of Harper and Pfenning [2005] which considers approximate shape of types and has been used to prove completeness of equivalence algorithm for LF's type theory cannot be used in our setting. Instead, we follow recent work by Abel and Scherer [2012a] and Abel et al. [2018] on proving normalization of our fully dependent type theory using a Kripke logical relation. Our semantic model highlights the intensional character of the LF function space and the extensional character of computations. Our main contribution is the design of the Kripke-style model for the dependent type theory Cocon that allows us to establish normalization. From the normalization proof, we derive type uniqueness, subject reduction, and consistency.

We believe Cocon lays the foundation to incorporate the methodology of logical frameworks into systems such as Agda [Norell 2007] or Coq [Bertot and Castéran 2004]. This finally allows us to combine the world of type theory and logical frameworks inheriting the best of both worlds.

1.4 Outline of the Technical Development

Before delving into the technical details, we sketch here the main structure of the technical development. Cocon consists of two mutually defined layers: LF to define HOAS and computation to write recursive programs. The syntax and typing rules of Cocon together with definitional equality are described in Sec. 2. We distinguish between two different kinds of variables, LF variables and computation variables. In particular, we define two different substitution operations. We then proceed to prove some elementary properties about LF (Sec. 3.1) and computation (Sec. 3.2), in particular well-formedness of LF contexts, LF Weakening and LF Substitution properties. For LF we also establish functionality of LF typing from which injectivity of LF function types follows.

Similarly, we establish some elementary properties about computation-level contexts and computation-level substitutions. We then proceed to define weak head reductions for LF and computations (Sec. 4) and show that they are closed under weakening (renaming).

Using weak head reduction, we define semantic equality using a Kripke model (Sec. 5). Our model is Kripke-style in the sense that it is closed under weakening. It contains all well-typed terms in weak head normal form (whnf) and is built on top of definitional equality. We do not define semantic typing, but say a term is semantically well typed, if it is semantically equal to itself. Since we embed computations inside LF terms, our typing rules for LF and computations are mutually defined, and one might wonder how we can break this cycle to arrive at a well-founded definition of semantic equality. We consider two LF terms M and N that weak head reduce to a $\lfloor t \rfloor_\sigma$ and $\lfloor t' \rfloor_{\sigma'}$ resp. semantically equal, if the computations t and t' are definitionally equal and the corresponding LF substitutions are semantically equal. This allows us to first define semantic equality for LF objects and subsequently semantic equality for computations breaking the cycle.

As we allow type-level computation, semantic equality for computations cannot be inductively defined on the structure of computation-level types. Instead, we use semantic kinding for types as a measure to define the semantic equality for computations.

Our semantic equality definitions are stable under renaming (weakening). We also prove symmetry, transitivity and type conversion for semantic equality that are the cornerstone of the development. This allows us to show that our semantic definition for terms is backwards closed and that neutral terms are semantically equal (see Sec. 6). Using the Kripke-model, we then show normalization and subject reduction (see Sec. 7). Logical consistency follows. The full development including the proofs can be found in the accompanying long version.

Summary of Contributions.

- We describe Cocon, a Martin-Löf style type theory with an an infinite hierarchy of universes and two intertwined layers: on the LF layer, we can define HOAS trees referring to values produced by computations and on the computation layer we can write recursive functions on HOAS trees and exploit the power of large eliminations. We mediate between these layers using contextual modal types. This allows us to bridge the gap between the intensional LF function space and the extensional function space used for writing recursive computations.
- We give a Kripke-style model to describe semantic equality for well-typed LF objects and well-typed computations highlighting the difference between intensional and extensional functions. Using this model we prove normalization.

2 COCON: COMPUTATION IN CONTEXTUAL TYPE THEORY

Cocon combines the logical framework LF [Harper et al. 1993] with a full dependent type theory that supports recursion over HOAS objects and universes. For clarity, we split Cocon's grammar into different syntactic categories (see Fig. 1). The LF layer describes LF objects, LF types, LF contexts; the computation layer consists of terms and types that describe recursive computation and universes. We mediate the interaction between LF objects and computations via a (contextual) box modality following Pientka [2008]: we embed contextual LF objects into computations, by pairing an LF object with its LF contexts and we embed computations within LF objects by unboxing the result of a computation. This allows us to not only write functions about LF objects, but also establish proofs about such functions and opens the way for metaprogramming and writing programs using reflection.

LF Kinds	K	::=	$\mathsf{type} \mid \Pi x{:}A.K$
LF Types	A, B	::=	$\mathsf{a}\ M_1 \ldots M_n \mid \Pi x{:}A.B$
LF Terms	M, N	::=	$\lambda x.M \mid x \mid \mathsf{c} \mid M\ N \mid \lfloor t \rfloor_\sigma$
LF Constants	a, c	::=	$\mathsf{tm} \mid \mathtt{lam} \mid \mathsf{app}$
LF Contexts	Ψ, Φ	::=	$\psi \mid \cdot \mid \Psi, x{:}A$
LF Context (Erased)	$\hat{\Psi}, \hat{\Phi}$	::=	$\psi \mid \cdot \mid \hat{\Psi},\ x$
LF Substitutions	σ	::=	$\cdot \mid \mathsf{wk}_{\hat{\Psi}} \mid \sigma, M$
LF Signature	Σ	::=	$\mathsf{tm}{:}\mathsf{type},\ \mathtt{lam}\,{:}\Pi y{:}(\Pi x{:}\mathsf{tm}.\mathsf{tm}).\mathsf{tm},\ \mathsf{app}{:}\Pi x{:}\mathsf{tm}.\Pi y{:}\mathsf{tm}.\mathsf{tm}$
Contextual Types	T	::=	$\Psi \vdash A \mid \Psi \vdash_\# A$
Contextual Objects	C	::=	$\hat{\Psi} \vdash M$
Sorts	u	::=	U_k
Domain of Discourse	$\check{\tau}$	::=	$\tau \mid \mathsf{tm\_ctx}$
Types and	$\tau, \mathcal{I},$	::=	$u \mid \lceil T \rceil \mid (y : \check{\tau}_1) \Rightarrow \tau_2$
Terms	t, s		$\mid y \mid \lceil C \rceil \mid t_1\ t_2 \mid \mathsf{fn}\ y \Rightarrow t \mid \mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t$
Branches	$\mathcal{B}$	::=	$(\psi, p \Rightarrow t_v \mid \psi, m, n, f_m, f_n \Rightarrow t_{\mathsf{app}} \mid \psi, m, f_m \Rightarrow t_{\mathtt{lam}})$
			$\mid (\psi \Rightarrow t_x \mid \psi, y, f_y \Rightarrow t_y)$
Contexts	Γ	::=	$\cdot \mid \Gamma, y : \check{\tau}$

Fig. 1. Syntax of Cocon

2.1 Syntax

Logical Framework LF with Embedded Computations. As in LF we allow dependent kinds and types; LF terms can be defined by LF variables, constants, LF applications, and LF lambda-abstractions. In addition, we allow a computation t to be embedded into LF terms using a closure $\lfloor t \rfloor_\sigma$. Here the computation t eventually computes to a *contextual* object that depends on assumptions Ψ following Pientka [2008]. Once computation of t produces a contextual object $\hat{\Psi} \vdash M$ we can embed the result by applying the substitution σ to M moving M from the LF context Ψ to the current context Φ.

We distinguish between computations that characterize a general LF term M of type A in a context Ψ using the contextual type $\Psi \vdash A$ and computations that are guaranteed to return a variable in a context Ψ of type A using the contextual type $\Psi \vdash_{\#} A$. This is essential when describing recursors over contextual LF terms, but also generally important when mechanizing formal systems and it is smoothly integrated in our type theory.

For simplicity, we fix here the LF signature to include the type tm and the LF constants `lam` and `app` . This allows us to for example define recursors on tm-objects directly.

LF contexts. LF contexts are either empty or are built by extending a context with a declaration $x{:}A$. We may also use a (context) variable ψ that stands for a context prefix and must be declared on the computation-level. In particular, we can write functions where we abstract over (context) variables. Consequently, we can pass LF contexts as arguments to functions. We classify LF contexts via schemas – for this paper, we pre-define the schema $\mathsf{tm\_ctx}$ which classifies a LF context which consists of tm declarations. Such context schemas are similar to adding base types to computation-level types. We often do not need to carry the full LF context with the type annotations, but it suffices to simply consider the erased LF context. Erased LF contexts are simply a list of variables possibly with a context variable at the head.

At the moment, we do not support computation on context at the moment – this simplifies the design. Recall that the head of a context denotes a possibly empty sequence of declarations. This prefix should be abstract and opaque to any LF term or LF type that is considered within this context. In other words, a LF term M (or LF type A) should be meaningful without requiring any specific knowledge about the prefix of declarations. Second, it would be difficult to enforce well-scoping and α-renaming. To illustrate, consider the following LF term $\mathsf{app}\ x\ y$ in the LF context $x{:}\mathsf{tm}, y{:}\mathsf{tm}$. If we were to allow type checking to exploit equivalence relations on LF contexts that take into account computations on LF contexts, we can argue that since $x{:}\mathsf{tm}, y{:}\mathsf{tm}$ is equivalent to $\mathsf{copy}\ \lceil x{:}\mathsf{tm}, y{:}\mathsf{tm}\rceil$, $\mathsf{app}\ x\ y$ should also be meaningful in the latter LF context. However, now the LF variables x and y are free in $\mathsf{app}\ x\ y$.

LF Substitutions. LF substitutions allow us to move between LF contexts. The empty LF substitution provides a mapping from an empty LF context to a LF context Ψ and hence has weakening built-in. The weakening substitution written as $\mathsf{wk}_{\hat{\Psi}}$ where Ψ describes the prefix of the range that corresponds to the domain; in other words it describes the weakening of the domain Ψ to the range $\Psi, \overrightarrow{x{:}A}$. In general, we may weaken any given LF context with the declarations $\overrightarrow{x{:}A}$. The generality of weakening substitutions is necessary to, for example, express that we can weaken a LF context ψ. We may write simply id, if $|\overrightarrow{x{:}A}| = 0$.

Weakening substitutions do not subsume the empty substitutions – only the empty substitution that maps the empty context to a concrete context $x_n{:}A_n, \ldots, x_1{:}A_1$ can be expressed as $\mathsf{wk}_{\cdot}$ where we annotate the weakening substitution with the empty LF context. For example, we would not be able to represent a substitution with the empty context as the domain and a context variable ψ as the range using a weakening substitution. Our built-in weakening substitutions are also sometimes called *renamings* as they only allow contexts to be extended to the right but they do not support arbitrary weakening of a LF context where we would insert a declaration in the middle (i.e. given a context $x{:}A_1, y{:}A_3$ we can weaken it to $x{:}A_1, w{:}A_2, y{:}A_3$).

From a de Bruijn perspective, the weakening substitution $\mathsf{wk}_{\cdot}$ which maps the empty context to $x_n{:}A_n, \ldots, x_1{:}A_1$ can be viewed as a shift n. Further, as in the de Bruijn world, $\mathsf{wk}_{x_n:A_n, \ldots, x_1:A_1}$ can be expanded and is equivalent to $\cdot, x_n, \ldots, x_1$. While our theory lends itself to an implementation with de Bruijn indices, we formulate our type theory using a named representation of variables. This not only simplifies our subsequent definitions of substitutions, but also leaves open how variables are realized in an implementation.

LF substitutions can also be built by extending a LF substitution σ with a LF term M. Following Nanevski et al. [2008], we do not store the domain of a substitution, but simply write them as a list of terms. We resurrect the domain of the substitution before applying it by erasing types from a context. To apply a substitution σ to a term M in an erased context $\hat{\Psi}$, we write $[\sigma/\hat{\Psi}]M$.

Contextual Objects and Types. We mediate between the LF and computation level using contextual types. We consider here general contextual LF terms that have type $\Psi \vdash A$, and contextual variable objects that have type $\Psi \vdash_{\#} A$.

Computations and their Types. Computations are formed by computation-level functions, written as $\mathsf{fn}\ y \Rightarrow t$, that are extensional, i.e. we can only observe their behaviour, applications, written as $t_1\ t_2$, boxed contextual objects, written as $\lceil C\rceil$, and recursor, written as $\mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t$. We annotate the recursor with the typing invariant $\mathcal{I}$ and recurse over the values computed by the term t. The LF context Ψ describes the local LF-world in which the value computed by t makes sense. Finally, $\mathcal{B}$ describes the different branches we can take depending on the value computed by t. These branches can be generated generically following Pientka and Abel [2015]. We focus on in the rest of the paper on the iterator over contextual objects of type $\lceil \Psi \vdash \mathsf{tm} \rceil$. In this case, we consider three different branches: 1) In the LF variable case, $(\psi, p \Rightarrow t_v)$, the variable p stands for a LF variable tm in the LF context ψ and has type $\lceil \psi \vdash_\# \mathsf{tm} \rceil$ and t_v is the body of the branch. 2) In the app -case, written as $(\psi, m, n, f_m, f_n \Rightarrow t_{\mathsf{app}})$, we pattern match on a LF term app $m\ n$ in the LF context ψ. The recursive calls are denoted by f_m and f_n respectively and t_{app} describes the body of the branch. 3) In the lam -case, written as $(\psi, m, f_m \Rightarrow t_{\mathtt{lam}})$, we pattern match on a LF term $\mathtt{lam}\ \lambda x.m$ where m denotes a LF term of LF type tm in the LF context $\psi, x{:}\mathsf{tm}$. The recursive call is described by f_m and the body of the branch is denoted by $t_{\mathtt{lam}}$.

For illustration, we also include other branches to construct also recursors over contextual objects of type $\lceil \Psi \vdash_\# \mathsf{tm} \rceil$, i.e. variables of type tm in the LF context Ψ. In this case, we only consider two branches where $\Psi = \Psi', x{:}\mathsf{tm}$: in the first branch $(\psi \Rightarrow t_x)$ we pattern match against x and ψ will be instantiated with Ψ'; in the second branch, $(\psi, y, f_y \Rightarrow t_y)$, the LF variable we are looking for is not x, but is somewhere in Ψ'. In this case, y denotes intuitively a LF variable that is not x and has type $\lceil \psi \vdash_\# \mathsf{tm} \rceil$ and we will instantiate ψ with Ψ'; f_y is the recursive call on the smaller LF context Ψ' and t_y is the body of the branch. We also include branches for recursing over LF substitution which have either the contextual type $\lceil \Psi \vdash \Phi \rceil$ or $\lceil \Psi \vdash_\# \Phi \rceil$. Here we consider two cases: either the LF substitution is empty then we choose the first branch, or it is of the shape σ, m and f_σ denotes the recursive call on the smaller LF substitution σ.

Computation-level types consist of boxed contextual types, written as $\lceil T \rceil$, and dependent types, written as $(y : \check{\tau}_1) \Rightarrow \tau_2$. We overload the dependent function space and allow as domain of discourse both computation-level types and the schema tm_ctx of LF context. To form both functions we use $\mathsf{fn}\ y \Rightarrow t$. We also overload function application $t\ s$ to eliminate dependent types $(y : \tau_1) \Rightarrow \tau_2$ and and $(y : \mathsf{tm\_ctx}) \Rightarrow \tau_2$, although in the latter case s stands for a LF context.

Cocon is a pure type system (PTS) with infinite hierarchy of predicative universes, written as U_k where $k \in \mathsf{Nat}$. The universes are not cumulative. We use sorts $u, k \in \mathcal{S}$, axioms $\mathfrak{A} = \{(\mathsf{U}_i,\ \mathsf{U}_{i+1} \mid i \in \mathsf{Nat}\}$, and rules $\mathfrak{R} = \{(\mathsf{U}_i,\ \mathsf{U}_j,\ \mathsf{U}_{\max(i,j)}) \mid i, j \in \mathsf{Nat}\}$. Universes add additional power.

Example 2.1. To illustrate the syntax of Cocon, we write a program that counts the number of constructors in a given tm. The type of the function is $\mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (m : \lceil \psi \vdash \mathsf{tm} \rceil) \Rightarrow \mathsf{nat}$.

$$\begin{array}{llcl}
\mathsf{fn}\ \psi \Rightarrow \mathsf{fn}\ m \Rightarrow \mathsf{rec}^{\mathcal{I}}\ (& \psi, p & \Rightarrow & 0 \\
 & \mid \psi, m, n, f_n, f_m & \Rightarrow & f_n + f_m + 1 \\
 & \mid \psi, m, f_m & \Rightarrow & f_m\ +\ 1\ )\ \psi\ m
\end{array}$$

The first branch describes the variable case where p describes a variable from the LF context ψ which has type $\lceil \psi \vdash_\# \mathsf{tm} \rceil$. The second branch describes the application case; here f_n and f_m respectively denote the recursive calls and have type nat. The third branch describes the lambda case where f_m is the recursive call made on the body of the lambda-term.

Example 2.2. Next we implement copy of type $\mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (m : \lceil \psi \vdash \mathsf{tm} \rceil) \Rightarrow \lceil \psi \vdash \mathsf{tm} \rceil$. We abbreviate the identity substitution wk_ψ by simply writing id.

$$\begin{array}{llcl}
\mathsf{fn}\ \psi \Rightarrow \mathsf{fn}\ m \Rightarrow \mathsf{rec}^{\mathcal{I}}\ (& \psi, p & \Rightarrow & \lceil \psi \vdash \lfloor p \rfloor_{\mathsf{id}}\ \rceil \\
 & \mid \psi, m, n, f_n, f_m & \Rightarrow & \lceil \psi \vdash \mathsf{app}\ \lfloor f_n \rfloor_{\mathsf{id}}\ \lfloor f_m \rfloor_{\mathsf{id}}\ \rceil \\
 & \mid \psi, m, f_m & \Rightarrow & \lceil \psi \vdash \mathtt{lam}\ \lambda x.\lfloor f_m \rfloor_{\mathsf{id}}\ \rceil\ )\ \psi\ m
\end{array}$$

In this example the input and output type depends on ψ; in particular the type of the recursive call f_m in the lambda case will be $\lceil \psi, x{:}\mathsf{tm} \vdash \mathsf{tm} \rceil$.

Example 2.3. We return the position of a LF variable in a LF context by writing a function pos that has type $\mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (x : \lceil \psi \vdash_\# \mathsf{tm} \rceil) \Rightarrow \mathsf{nat}$.

$$\begin{array}{llcl}
\mathsf{fn}\ \psi \Rightarrow \mathsf{fn}\ x \Rightarrow \mathsf{rec}^{\mathcal{I}}\ (& \psi & \Rightarrow & 0 \\
 & \mid \psi, y, f_y & \Rightarrow & 1 + f_y)\ \psi\ x
\end{array}$$

2.2 LF Substitution Operation

Our type theory distinguishes between LF-variables and computation-level variables. We have substitution operation for both. Let's consider first a few examples to get a better intuition. Let's look at a few examples to get a better intuition.

Examples 1. Consider the LF term $\mathsf{app}\ \lfloor \lceil x, y \vdash \mathsf{app}\ x\ y \rceil \rfloor_{\mathsf{wk}_{x,y}}\ w$. This LF term is obviously well-typed in the (normal) LF context $x{:}\mathsf{tm}, y{:}\mathsf{tm}, w{:}\mathsf{tm}$ and applying the substitution $\mathsf{wk}_{x,y}$ to $\mathsf{app}\ x\ y$ is meaningful as $\mathsf{wk}_{x,y}$ expands to $\cdot, x, y$. When we apply $\cdot, x, y$ to unbox $\lceil x, y \vdash \mathsf{app}\ x\ y \rceil$, we resurrect the domain and apply $[\cdot, x, y\ /\ \cdot, x, y](\mathsf{app}\ x\ y)$.

Examples 2. What about considering the α-equivalent term $\lfloor \lceil x', y' \vdash \mathsf{app}\ x'\ y' \rceil \rfloor_{\mathsf{wk}_{x,y}}$? – Again we observe that $\mathsf{wk}_{x,y}$ expands to $\cdot, x, y$; when we apply $\cdot, x, y$ to unbox $\lceil x', y' \vdash \mathsf{app}\ x'\ y' \rceil$, we resurrect the domain and apply $[\cdot, x, y\ /\ \cdot, x', y'](\mathsf{app}\ x'\ y')$ effectively renaming x' and y' to x and y respectively.

$$
\begin{array}{lll}
[\sigma/\hat{\Psi}](\lambda x.M) = \lambda x.M' & \text{where } [\sigma, x/\hat{\Psi}, x](M) = M' & \text{provided that } x \notin \mathsf{FV}(\sigma) \text{ and } x \notin \hat{\Psi} \\
[\sigma/\hat{\Psi}](M\ N) \;= M'\ N' & \text{where } [\sigma/\hat{\Psi}](M) = M' \text{ and } [\sigma/\hat{\Psi}](N) = N' & \\
[\sigma/\hat{\Psi}](\lfloor t \rfloor_{\sigma'}) \;= \lfloor t \rfloor_{\sigma''} & \text{where } [\sigma/\hat{\Psi}](\sigma') = \sigma'' & \\
[\sigma/\hat{\Psi}](x) \quad\;\; = M & \text{where lookup } x\ [\sigma/\hat{\Psi}] = M & \\
[\sigma/\hat{\Psi}]c \qquad\;\; = \;c & & \\
 & & \\
[\sigma/\hat{\Psi}](\cdot) \qquad = \;\cdot & & \\
[\sigma/\hat{\Psi}](\mathsf{wk}_{\hat{\Phi}}) \;= \sigma' & \text{where } \mathrm{trunc}_{\Phi}\ (\sigma/\hat{\Psi}) = \sigma' & \\
[\sigma/\hat{\Psi}](\sigma', M) = \sigma'', M' & \text{where } [\sigma/\hat{\Psi}](\sigma') = \sigma'' \text{ and } [\sigma/\hat{\Psi}](M) = M' &
\end{array}
$$

Fig. 2. Simultaneous LF Substitution for LF Objects

We define LF substitutions uniformly using simultaneous substitution operation written as $[\sigma/\hat{\Psi}]M$ (and similarly $[\sigma/\hat{\Psi}]A$ and $[\sigma/\hat{\Psi}]K$) (see Fig. 2). As LF substitutions are simply a list of terms, we need to resurrect the domain to lookup the instantiation for a LF variable x in σ. This is always possible. When pushing the substitution through an application $M\ N$, we simply apply it to M and N respectively. When pushing the LF substitution through a λ-abstraction, we extend it. When applying σ to a LF variable x, we retrieve the corresponding instantiation from σ using the auxiliary function lookup which works mostly as expected. When applying the LF substitution σ to the LF closure $\lfloor t \rfloor_{\sigma'}$ we leave t untouched, since t cannot contain any free LF variables and compose σ and σ'.

$$
\begin{array}{lll}
\text{lookup } x\ [\sigma, M/\hat{\Psi}, x] = M & & \\
\text{lookup } x\ [\sigma, M/\hat{\Psi}, y] = \text{lookup } x\ [\sigma/\hat{\Psi}] & & \\
\text{lookup } x\ [\mathsf{wk}_{\hat{\Psi}}/\hat{\Psi}] \quad = x & & \text{where } x \in \hat{\Psi} \\
\text{lookup } x\ [\sigma/\hat{\Psi}] \qquad = \text{fails otherwise} & &
\end{array}
$$

Composition of LF substitution is straightforward. When we apply σ to $\mathsf{wk}_{\hat{\Psi}}$, we truncate σ and only keep those entries corresponding to the LF context Ψ. Recall that $\mathsf{wk}_{\hat{\Psi}}$ provides a weakening substitution from a context Ψ to another context $\Psi, \overrightarrow{x{:}A}$ where $|\overrightarrow{x{:}A}| = n$. The simultaneous substitution σ provides mappings for all the variables in $\hat{\Psi}, \vec{x}$. The result of $[\sigma/\hat{\Psi}, \vec{x}]\mathsf{wk}_{\hat{\Psi}}$ then should only provide mappings for all the variables in Ψ. We use the operation trunc to remove irrelevant instantiations. The definition of truncation is straightforward.

$$
\begin{array}{ll}
\mathrm{trunc}_{\Psi}\ (\sigma/\hat{\Psi}) & = \sigma \\
\mathrm{trunc}_{\Psi}\ (\sigma, M/\hat{\Phi}, x) & = \mathrm{trunc}_{\Psi}\ (\sigma/\hat{\Phi}) \\
\mathrm{trunc}_{\Psi}\ (\mathsf{wk}_{(\hat{\Psi}, \vec{x})}/\hat{\Psi}, \vec{x}) & = \mathsf{wk}_{\hat{\Psi}} \\
\mathrm{trunc}_{\Psi}\ (\cdot/\cdot) & = \text{fails } \Psi \neq \cdot
\end{array}
$$

Computation-level Substitution for Terms

$$
\begin{array}{lll}
\{t/y\}(\mathsf{fn}\ x \Rightarrow t') & = \mathsf{fn}\ x \Rightarrow \{t/y\}t' & \text{provided } x \notin \mathsf{FV}(t) \\
\{t/y\}(t_1\ t_2) & = \{t/y\}t_1\ \{t/y\}t_2 & \\
\{t/y\}(\mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t) & = \mathsf{rec}^{\mathcal{I}}\ \{t/y\}b_1 \mid \{t/y\}b_k\ \{t/y\}\Psi\ \{t/y\}t & \text{where } \mathcal{B} = b_1 \mid \ldots \mid b_n \\
\{t/y\}\lceil C \rceil & = \lceil \{t/y\}(C) \rceil & \\
\{t/y\}(y) & = t & \\
\{t/y\}(x) & = x & \text{where } x \neq y \\
\{t/y\}(\cdot) & = \cdot & \\
\{t/y\}(\Psi, x : A) & = \{t/y\}\Psi, x : \{t/y\}A & \text{provided } (\hat{\Psi}, x) \notin \mathsf{FV}(t)
\end{array}
$$

Computation-level Substitution for Branches

$$
\begin{array}{lll}
\{t/y\}(\vec{x} \Rightarrow t') & = (\vec{x} \Rightarrow \{t/y\}t') & \text{provided } \vec{x} \notin \mathsf{FV}(t)
\end{array}
$$

Computation-level Substitution for Contextual Objects

$$
\begin{array}{lll}
\{t/y\}(\hat{\Psi} \vdash M) & = \{t/y\}\hat{\Psi} \vdash \{t/y\}M & \text{provided } \hat{\Psi} \notin \mathsf{FV}(t) \\
\{t/y\}(\hat{\Psi} \vdash \sigma) & = \{t/y\}\hat{\Psi} \vdash \{t/y\}\sigma & \text{provided } \hat{\Psi} \notin \mathsf{FV}(t)
\end{array}
$$

Computation-level Substitution for LF Objects

$$
\begin{array}{ll}
\{t/y\}(\lambda x.M) & = \lambda x.\{t/y\}M \\
\{t/y\}(M\ N) & = \{t/y\}M\ \{t/y\}N \\
\{t/y\}(\lfloor t' \rfloor_{\sigma}) & = \lfloor \{t/y\}t' \rfloor_{\{t/y\}\sigma} \\
\{t/y\}(\mathsf{c}) & = \mathsf{c} \\
\{t/y\}(x) & = x
\end{array}
$$

Computation-level Substitution for LF Substitutions

$$
\begin{array}{ll}
\{t/y\}(\cdot) & = \cdot \\
\{t/y\}(\sigma, M) & = \{t/y\}\sigma,\ \{t/y\}M \\
\{t/y\}(\mathsf{wk}_{\hat{\Psi}}) & = \mathsf{wk}_{\{t/y\}(\hat{\Psi})}
\end{array}
$$

Fig. 3. Computation-level Substitution

2.3 Computation-level Substitution Operation

The computation-level substitution operation $\{t/x\}t'$ traverses the computation t' and replaces any free occurrence of the computation-level variable x in t' with t(see Fig. 3). The interesting case is $\{t/x\}\lceil C \rceil$. Here we push the substitution into C and we will further apply it to objects in the LF layer. When we encounter a closure such as $\lfloor t'' \rfloor_{\sigma}$, we continue to push it inside σ and also into t''. When substituting a LF context Ψ for the variable ψ in a context Φ, we rename the declarations present in Φ. This is a convention. It would equally work to rename the variable declarations in Ψ. For example, in $\{(x{:}\mathsf{tm}, y{:}\mathsf{tm})/\psi\}(\hat{\psi}, x \vdash \mathtt{lam}\ \lambda y.\mathsf{app}\ x\ y\ )$, we rename the variable x in $\hat{\psi}, x$ and replace ψ with $(x{:}\mathsf{tm}, y{:}\mathsf{tm})$ in $(\hat{\psi}, w \vdash \mathtt{lam}\ \lambda y.\mathsf{app}\ w\ y)$. This results in $x, y, w \vdash \mathtt{lam}\ \lambda y.\mathsf{app}\ w\ y$. When type checking this term we will eventually also α-rename the λ-bound LF variable y.

2.4 LF Typing

We concentrate here on the typing rules for LF terms, LF substitutions and LF contexts (see Fig. 5). The rules for LF types and kinds are straightforward (see Fig. 4). All of the typing rules have access to a LF signature Σ which we omit to keep the presentation compact. In typing rules for LF abstractions $\lambda x.M$ we simply extend the LF context and check the body M. When we encounter a LF variable, we look up its type in the LF context. The conversion rule is important and subtle. We only allow conversion of types – conversion of the LF context is not necessary, as we do not allow computations to appear directly in the LF context and we can keep part of the LF context abstract. However, we deviate

$\boxed{\Gamma; \Phi \vdash A : \mathsf{type}}$ and $\boxed{\Gamma; \Phi \vdash K : \mathsf{kind}}$ LF type A is well-kinded and LF kind K is well-formed

$$\frac{\Gamma \vdash \Psi : \mathsf{ctx} \quad \mathsf{a}{:}K \in \Sigma}{\Gamma; \Psi \vdash \mathsf{a} : K} \quad \frac{\Gamma; \Psi \vdash P : \Pi x{:}A.K \quad \Gamma; \Psi \vdash M : A}{\Gamma; \Psi \vdash P\, M : [M/x]K} \quad \frac{\Gamma; \Psi \vdash A : \mathsf{type} \quad \Gamma; \Psi, x{:}A \vdash B : \mathsf{type}}{\Gamma; \Psi \vdash \Pi x{:}A.B : \mathsf{type}}$$

$$\frac{\Gamma; \Psi \vdash A : K' \quad \Gamma; \Psi \vdash K' \equiv K : \mathsf{kind}}{\Gamma; \Psi \vdash A : K} \quad \frac{\Gamma \vdash \Psi : \mathsf{ctx}}{\Gamma; \Psi \vdash \mathsf{type} : \mathsf{kind}} \quad \frac{\Gamma; \Psi \vdash A : \mathsf{type} \quad \Gamma; \Psi, x{:}A \vdash K : \mathsf{kind}}{\Gamma; \Psi \vdash \Pi x{:}A.K : \mathsf{kind}}$$

Fig. 4. Kinding Rules for LF Types

$\boxed{\Gamma; \Psi \vdash_{\#} M : A}$ LF term M of LF type A in the LF context Ψ and context Γ describes a variable

$$\frac{\Gamma; \Psi \vdash M \equiv x : A \quad \Psi(x) = A}{\Gamma; \Psi \vdash_{\#} M : A} \qquad \frac{\Gamma; \Psi \vdash_{\#} M : B \quad \Gamma; \Psi \vdash B \equiv A : \mathsf{type}}{\Gamma; \Psi \vdash_{\#} M : A}$$

$$\frac{\Gamma; \Psi \vdash M \equiv \lfloor t \rfloor_{\sigma} : [\sigma/\hat{\Phi}](A) \quad \Gamma \vdash t : [\Phi \vdash_{\#} A] \quad \Gamma; \Psi \vdash_{\#} \sigma : \Phi}{\Gamma; \Psi \vdash_{\#} M : A}$$

$\boxed{\Gamma; \Psi \vdash M : A}$ LF term M has LF type A in the LF context Ψ and context Γ

$$\frac{\Gamma; \Psi, x{:}A \vdash M : B}{\Gamma; \Psi \vdash \lambda x.M : \Pi x{:}A.B} \quad \frac{\Gamma \vdash t : [\Phi \vdash A] \text{ or } \Gamma \vdash t : [\Phi \vdash_{\#} A] \quad \Gamma; \Psi \vdash \sigma : \Phi}{\Gamma; \Psi \vdash \lfloor t \rfloor_{\sigma} : [\sigma/\hat{\Phi}]A}$$

$$\frac{\Gamma \vdash \Psi : \mathsf{ctx} \quad \Psi(x) = A}{\Gamma; \Psi \vdash x : A} \qquad \frac{\Gamma; \Psi \vdash M : B \quad \Gamma; \Psi \vdash B \equiv A : \mathsf{type}}{\Gamma; \Psi \vdash M : A}$$

$$\frac{\Gamma; \Psi \vdash M : \Pi x{:}A.B \quad \Gamma; \Psi \vdash N : A}{\Gamma; \Psi \vdash M\, N : [N/x]B} \quad \frac{\Gamma \vdash \Psi : \mathsf{ctx} \quad \mathsf{c} : A \in \Sigma}{\Gamma; \Psi \vdash \mathsf{c} : A}$$

$\boxed{\Gamma; \Phi \vdash \sigma : \Psi}$ LF substitution σ provides a mapping from the LF context Ψ to Φ

$$\frac{\Gamma \vdash \Psi, \overrightarrow{x{:}A} : \mathsf{ctx}}{\Gamma; \Psi, \overrightarrow{x{:}A} \vdash \mathsf{wk}_{\hat{\Psi}} : \Psi} \quad \frac{\Gamma \vdash \Phi : \mathsf{ctx}}{\Gamma; \Phi \vdash \cdot : \cdot} \quad \frac{\Gamma; \Phi \vdash \sigma : \Psi \quad \Gamma; \Phi \vdash M : [\sigma/\hat{\Psi}]A}{\Gamma; \Phi \vdash \sigma, M : \Psi, x{:}A}$$

$\boxed{\Gamma; \Psi \vdash_{\#} \sigma : \Phi}$ LF substitution σ from LF context Φ to the LF context Ψ is a weakening subst.

$$\frac{\Gamma; \Psi, \overrightarrow{x{:}A} \vdash \sigma \equiv \mathsf{wk}_{\hat{\Psi}} : \Psi}{\Gamma; \Psi, \overrightarrow{x{:}A} \vdash_{\#} \sigma : \Psi}$$

Fig. 5. Typing Rules for LF Terms and LF Substitutions

from Cave and Pientka [2012] in the rule that allows us to embed computations into LF terms. Given a computation t that has type $\lceil \Psi \vdash A \rceil$ or $\lceil \Psi \vdash_{\#} A \rceil$, we can embed it into the current LF context Φ by forming the closure $\lfloor t \rfloor_{\sigma}$ where σ provides a mapping for the variables in Ψ. This formulation generalizes previous work which only allowed variables declared in Γ to be embedded in LF terms. This enforced a strict separation between computations and LF terms. The typing rules for LF substitutions are as expected.

Last, we consider the typing rules for LF contexts (see Fig. 6). They simply analyze the structure of a LF context. When we reach the head we either encounter a empty LF context or an context variable y which must be declared in the computation-level context Γ.

$$\boxed{\Gamma \vdash \Psi : \mathsf{ctx}} \text{ LF context } \Psi \text{ is a well-formed}$$

$$\frac{\vdash \Gamma}{\Gamma \vdash \cdot : \mathsf{ctx}} \quad \frac{\Gamma(y) = \mathsf{tm\_ctx} \quad \vdash \Gamma}{\Gamma \vdash y : \mathsf{ctx}} \quad \frac{\Gamma \vdash \Psi : \mathsf{ctx} \quad \Gamma; \Psi \vdash A : \mathsf{type}}{\Gamma \vdash \Psi, x{:}A : \mathsf{ctx}}$$

Fig. 6. Typing Rules for LF Contexts

2.5 Definitional LF Equality

We now consider definitional LF equality. We omit the transitive closure rules as well as congruence rules, but concentrate here on the reduction and expansion rules. For LF terms, equality is $\beta\eta$. In addition, we can reduce $\lfloor \Psi \vdash M \rfloor_\sigma$ by simply applying σ to M.

For LF substitutions, we take into account that weakening substitutions are not unique. For example, the substitution $\mathsf{wk}_\cdot$ may stand for a mapping from the empty context to another LF context; so does the empty substitution $\cdot$. Similarly, $\mathsf{wk}_{x_1,\ldots x_n}$ is equivalent to $\mathsf{wk}_\cdot, x_1, \ldots, x_n$.

$$\boxed{\Gamma; \Psi \vdash M \equiv N : A} \quad \text{LF Term } M \text{ is definitionally equal to LF Term } N \text{ at LF type } A$$

$$\frac{\Gamma; \Psi \vdash M : \Pi x{:}A.B}{\Gamma; \Psi \vdash M \equiv \lambda x.M\ x : \Pi x{:}A.B} \quad \frac{\Gamma; \Psi, x{:}A \vdash M_1 : B \quad \Gamma; \Psi \vdash M_2 : A}{\Gamma; \Psi \vdash (\lambda x.M_1)\ M_2 \equiv [M_2/x]M_1 : [M_2/x]B}$$

$$\frac{\Gamma; \Phi \vdash N : A \quad \Gamma; \Psi \vdash \sigma : \Phi}{\Gamma; \Psi \vdash \lfloor \lceil \hat{\Phi} \vdash N \rceil \rfloor_\sigma \equiv [\sigma/\hat{\Phi}]N : [\sigma/\hat{\Phi}]A}$$

$$\boxed{\Gamma; \Psi \vdash \sigma \equiv \sigma' : \Phi} \quad \text{LF Substitution } \sigma \text{ is definitionally equal to LF Substitution } \sigma'$$

$$\frac{\Gamma \vdash \Psi : \mathsf{ctx}}{\Gamma; \Psi \vdash \mathsf{wk}_\cdot \equiv \cdot : \cdot} \quad \frac{\Gamma \vdash \Phi, x{:}A, \overrightarrow{y{:}B} : \mathsf{ctx}}{\Gamma; \Phi, x{:}A, \overrightarrow{y{:}B} \vdash \mathsf{wk}_{\hat{\Phi},x} \equiv \mathsf{wk}_\Phi, x : \Phi, x{:}A}$$

$$\frac{\Gamma; \Psi \vdash \sigma \equiv \sigma' : \Phi \quad \Gamma; \Psi \vdash M \equiv N : [\sigma/\hat{\Phi}]A}{\Gamma; \Psi \vdash \sigma, M \equiv \sigma', N : \Phi, x{:}A}$$

Fig. 7. Reduction and Expansion for LF Terms and LF Substitutions

2.6 Contextual LF Typing and Definitional Equivalence

We describe typing and equivalence of contextual objects in Fig. 8. This is standard. We lift definitional equality on LF terms to contextual objects. Note that we overload notation, writing $\hat{\Psi}$ for a LF context Ψ where we have already erased type declarations, but we sometimes abuse notation and write $\hat{\Psi}$ for taking a LF context Ψ and erasing its type information.

2.7 Computation Typing

We describe well-typed computations in Fig. 9 using the typing judgment $\Gamma \vdash t : \tau$. Computations only have access to computation-level variables declared in the context Γ. To avoid duplication of typing rules, we overload the typing judgment and write $\check{\tau}$ instead of τ, if the same judgment is used to check that a given LF context is of schema $\mathsf{tm\_ctx}$. For example, to ensure that $(y : \check{\tau}_1) \Rightarrow \tau_2$ has kind u_3, we check that $\check{\tau}_1$ is well-kinded. For compactness, we abuse notation writing $\Gamma \vdash \mathsf{tm\_ctx} : u$ although the schema $\mathsf{tm\_ctx}$ is not a proper type whose elements can be computed. In the typing rules for computation-level (extensional) functions, the input to the function which we also call domain of discourse may either be of type τ_1 or $\mathsf{tm\_ctx}$. To eliminate a term t of type $(y : \tau_1) \Rightarrow \tau_2$, we check that s is of type τ_1 and then return $\{s/y\}\tau_2$ as the type of $t\ s$. To eliminate a term of type $(y : \mathsf{tm\_ctx}) \Rightarrow \tau$, we overload application

$\boxed{\Gamma \vdash T}$ Contextual Type T is well-kinded

$$\frac{\Gamma; \Psi \vdash A : \mathsf{type}}{\Gamma \vdash (\Psi \vdash A)} \quad \frac{\Gamma; \Psi \vdash A : \mathsf{type}}{\Gamma \vdash (\Psi \vdash_{\#} A)}$$

$\boxed{\Gamma \vdash C : T}$ Contextual Objects C has Contextual Type T in context Γ

$$\frac{\Gamma; \Psi \vdash M : A}{\Gamma \vdash (\hat{\Psi} \vdash M) : (\Psi \vdash A)} \quad \frac{\Gamma; \Psi \vdash_{\#} M : A}{\Gamma \vdash (\hat{\Psi} \vdash M) : (\Psi \vdash_{\#} A)}$$

$\boxed{\Gamma \vdash C \equiv C' : T}$ Definitional Equivalence between Contextual Object

$$\frac{\Gamma; \Psi \vdash M \equiv N : A}{\Gamma \vdash (\hat{\Psi} \vdash M) \equiv (\hat{\Psi} \vdash N) : (\Psi \vdash A)}$$

$\boxed{\Gamma \vdash T \equiv T'}$ Definitional Equivalence between Contextual Types

$$\frac{\Gamma \vdash \Psi \equiv \Phi : \mathsf{ctx} \quad \Gamma; \Psi \vdash A \equiv B : \mathsf{type}}{\Gamma \vdash (\Psi \vdash A) \equiv (\Phi \vdash B)}$$

$$\frac{\Gamma \vdash \Psi \equiv \Phi : \mathsf{ctx} \quad \Gamma; \Psi \vdash A \equiv B : \mathsf{type}}{\Gamma \vdash (\Psi \vdash_{\#} A) \equiv (\Phi \vdash_{\#} B)}$$

Fig. 8. Typing and Equivalence Rules for Contextual Objects

simply writing $t\ s$, although s stands for a LF context and check that s is of schema $\mathsf{tm\_ctx}$. This distinction between the domains of discourse is important, as we only allow LF contexts to be built either by a context variable or a LF type declaration. We can embed contextual object C into computations by boxing it and transitioning to the typing rules for LF. We eliminate contextual types using the recursor.

In general, the output type of the recursor may depend on the argument we are recursing over. We hence annotate the recursor itself with an invariant $\mathcal{I}$. We consider only the recursor for contextual LF terms where $\mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (y : \lceil \psi \vdash \mathsf{tm} \rceil) \Rightarrow \tau$, but other recursors follow similar ideas. To check that the recursor $\mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t$ has type $\{\Psi/\psi, t/y\}\tau$, we check that each of the three branches has the specified type $\mathcal{I}$. In the base case, we may assume in addition to $\psi : \mathsf{tm\_ctx}$ that we have a variable $p : \lceil \psi \vdash_{\#} \mathsf{tm} \rceil$ and check that the body has the appropriate type. If we encounter a contextual LF object built with the LF constant app , then we choose the branch b_{app} . We assume $\psi : \lceil \mathsf{tm\_ctx} \rceil$, $m : \lceil \psi \vdash \mathsf{tm} \rceil$, $n : \lceil \psi \vdash \mathsf{tm} \rceil$, as well as f_n and f_m which stand for the recursive calls on m and n respectively. We then check that the body t_{app} is well-typed. If we encounter a LF object built with the LF constant $\mathtt{lam}$, then we choose the branch $b_{\mathtt{lam}}$. We assume $\psi : \lceil \mathsf{tm\_ctx} \rceil$ and $m : \lceil \psi, x{:}\mathsf{tm} \vdash \mathsf{tm} \rceil$ together with the recursive call f_m on m in the extended LF context $\psi, x{:}\mathsf{tm}$. We then check that the body $t_{\mathtt{lam}}$ is well-typed.

2.8 Definitional Equality for Computations

We now consider definitional equality for computations concentrating on the reduction rules. We omit the transitive closure and congruence rules, as they are as expected.

We consider two computations to be equal, if they evaluate to the same result. We propagate values through computations and types relying on the computation-level substitution operation. When we apply a term s to a computation $\mathsf{fn}\ y \Rightarrow t$, we β-reduce and replace y in the body t with s. We unfold the recursor depending on the value passed. If it is $\lceil \hat{\Psi} \vdash \mathtt{lam}\, \lambda x.M \rceil$, then we choose the branch $t_{\mathtt{lam}}$. If the value is $\lceil \hat{\Psi} \vdash \mathsf{app}\ M\ N \rceil$, we continue with the branch t_{app} . If it is $\lceil \hat{\Psi} \vdash x \rceil$, i.e. the variable case, we continue with t_v. Note that if Ψ is empty, then the case for variables is unreachable, since there is no LF variable of type tm in the empty LF context and hence the contextual type $\lceil \cdot \vdash_{\#} \mathsf{tm} \rceil$ is empty.

We also include the expansion of a computation t at type $\lceil \Psi \vdash A \rceil$; it is equivalent to unboxing t with the identity substitution and subsequently boxing it, i.e. t is equivalent to $\lceil \hat{\Psi} \vdash \lfloor t \rfloor_{\mathsf{wk}_{\hat{\Psi}}} \rceil$.

Well-formed Context: $\boxed{\vdash \Gamma}$ $\qquad \dfrac{}{\vdash \cdot} \qquad \dfrac{\vdash \Gamma \quad \Gamma \vdash \check{\tau} : u}{\vdash \Gamma, x{:}\check{\tau}}$

Typing and Kinding Judgments for Computations $\boxed{\Gamma \vdash t : \tau}$ and $\boxed{\Gamma \vdash \tau : u}$

$$\frac{\vdash \Gamma}{\Gamma \vdash u_1 : u_2}\ (u_1, u_2) \in \mathfrak{A} \qquad \frac{\Gamma \vdash \check{\tau}_1 : u_1 \quad \Gamma, y{:}\tau_1 \vdash \tau_2 : u_2}{\Gamma \vdash (y : \check{\tau}_1) \Rightarrow \tau_2 : u_3}\ (u_1,\ u_2,\ u_3) \in \mathfrak{R} \qquad \frac{\Gamma \vdash T}{\Gamma \vdash \lceil T \rceil : u}$$

$$\frac{y : \check{\tau} \in \Gamma \quad \vdash \Gamma}{\Gamma \vdash y : \check{\tau}} \qquad \frac{\Gamma \vdash t : (y : \check{\tau}_1) \Rightarrow \tau_2 \quad \Gamma \vdash s : \check{\tau}_1}{\Gamma \vdash t\ s : \{s/y\}\tau_2} \qquad \frac{\Gamma, y : \check{\tau}_1 \vdash t : \tau_2 \quad \Gamma \vdash (y : \check{\tau}_1) \Rightarrow \tau_2 : u}{\Gamma \vdash \mathsf{fn}\ y \Rightarrow t : (y : \check{\tau}_1) \Rightarrow \tau_2}$$

$$\frac{\Gamma \vdash C : T}{\Gamma \vdash \lceil C \rceil : \lceil T \rceil} \qquad \frac{\Gamma \vdash t : \tau' \quad \Gamma \vdash \tau' \equiv \tau : u}{\Gamma \vdash t : \tau}$$

Schema checking of LF Context $\boxed{\Gamma \vdash \Psi : \mathsf{tm\_ctx}}$;Well-Formedness of Schema $\boxed{\Gamma \vdash \mathsf{tm\_ctx} : u}$

$$\frac{\vdash \Gamma}{\Gamma \vdash \mathsf{tm\_ctx} : u} \qquad \frac{\vdash \Gamma}{\Gamma \vdash \cdot : \mathsf{tm\_ctx}} \qquad \frac{\Gamma \vdash \Psi : \mathsf{tm\_ctx} \quad \Gamma; \Psi \vdash A : \mathsf{type} \quad \Gamma; \Psi \vdash A \equiv \mathsf{tm} : \mathsf{type}}{\Gamma \vdash \Psi, x{:}A : \mathsf{tm\_ctx}}$$

Fig. 9. Typing Rules for Computations (without recursor)

3 ELEMENTARY PROPERTIES OF TYPING AND DEFINITIONAL EQUALITY

We now state and prove some basic properties about our type theory before we give its semantic interpretation and show that all well-typed terms normalize. For stating the theorems succinctly, we refer to judgments that only depend on the computation context Γ using $\mathcal{J}_{\mathsf{comp}}$ and judgments that refer to both the computation context Γ and the LF context Ψ with $\mathcal{J}_{\mathsf{LF}}$.

$$\begin{array}{lcl} \mathcal{J}_{\mathsf{comp}} & = & \{t : \check{\tau}, t \equiv t' : \check{\tau}\} \\ \mathcal{J}_{\mathsf{LF}} & = & \{M : A, M \equiv N : A, A : K, A \equiv B : K, K : \mathsf{kind}, K \equiv K' : \mathsf{kind}, \sigma : \Psi, \sigma \equiv \sigma' : \Psi\} \end{array}$$

Next we prove some elementary properties for LF and computations. As we separate the LF variables from the computation-level variables, we establish first properties such as well-formedness of context, weakening and substitution, for LF and then we prove the dual properties for computations.

3.1 Elementary Properties of LF

THEOREM 3.1 (WELL-FORMEDNESS OF LF CONTEXT).

(1) If $\mathcal{D} :: \Gamma \vdash \Psi, x{:}A, \Psi' : \mathsf{ctx}$ then $\mathcal{C} ::\ \Gamma \vdash \Psi : \mathsf{ctx}$ and $\mathcal{C} < \mathcal{D}$.
(2) If $\mathcal{D} :: \Gamma; \Psi \vdash \mathcal{J}_{\mathsf{LF}}$ then $\mathcal{C} ::\ \Gamma \vdash \Psi : \mathsf{ctx}$ and $\mathcal{C} < \mathcal{D}$.

PROOF. 1(1) by induction on the structure of Ψ'; 2(2) by induction on $\Gamma; \Psi \vdash \mathcal{J}_{\mathsf{LF}}$.

First statement: If $\mathcal{D} :: \Gamma \vdash \Psi, x{:}A, \Psi'$ then $\mathcal{C} ::\ \Gamma \vdash \Psi$ and $\mathcal{C} < \mathcal{D}$

Case. $\Psi' = \cdot$

$\mathcal{D} :: \Gamma \vdash \Psi, x{:}A : \mathsf{ctx}$	by assumption
$\mathcal{C} :: \Gamma \vdash \Psi : \mathsf{ctx}$ and $\mathcal{C} < \mathcal{D}$	by inversion

Case. $\Psi' = \Psi'', x' : B$

$\mathcal{D} :: \Gamma \vdash \Psi, x{:}A, \Psi'', x'{:}B : \mathsf{ctx}$	by assumption
$\mathcal{D} :: \Gamma \vdash \Psi, x{:}A, \Psi''$ and $\mathcal{D}' < \mathcal{C}'$	by inversion
$\mathcal{C} :: \Gamma \vdash \Psi$ and $\mathcal{C} < \mathcal{D}$	by IH

Recursor over LF Parameters $\mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (q : \lceil \psi \vdash_{\#} \mathsf{tm} \rceil) \Rightarrow \tau$

$$\frac{\Gamma \vdash t : \lceil \Psi \vdash_{\#} \mathsf{tm} \rceil \quad \Gamma \vdash \mathcal{I} : u \quad \Gamma \vdash (\psi \Rightarrow b_e) : \mathcal{I} \quad \Gamma \vdash (\psi, q, f_q \Rightarrow b_c) : \mathcal{I}}{\Gamma \vdash \mathsf{rec}^{\mathcal{I}} (\psi \Rightarrow b_e \mid \psi, q, f_q \Rightarrow b_c)\ \Psi\ t : \{\Psi/\psi,\ t/y\}\tau}$$

Branches where $\quad \mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (y : \lceil \psi \vdash_{\#} \mathsf{tm} \rceil) \Rightarrow \tau$

$$\frac{\Gamma, \psi : \mathsf{tm\_ctx} \vdash b_e : \{(\psi, x{:}\mathsf{tm})/\psi,\ \lceil \psi, x \vdash x \rceil / p\}\tau}{\Gamma \vdash (\psi \Rightarrow b_e) : \mathcal{I}}$$

$$\frac{\Gamma, \psi : \mathsf{tm\_ctx}, q : \lceil \psi \vdash_{\#} \mathsf{tm} \rceil, f_q : \{q/p\}\tau \vdash b_c : \{(\psi, x{:}\mathsf{tm})/\psi,\ \{\lceil \psi, x \vdash \lfloor q \rfloor_{\mathsf{wk}_\psi} \rceil / p\}\tau}{\Gamma \vdash (\psi, q, f_q \Rightarrow b_c) : \mathcal{I}}$$

Recursor over LF Terms $\mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (y : \lceil \psi \vdash \mathsf{tm} \rceil) \Rightarrow \tau$

$$\frac{\Gamma \vdash t : \lceil \Psi \vdash \mathsf{tm} \rceil \quad \Gamma \vdash \mathcal{I} : u \quad \Gamma \vdash b_v : \mathcal{I} \quad \Gamma \vdash b_{\mathsf{app}} : \mathcal{I} \quad \Gamma \vdash b_{\mathtt{lam}} : \mathcal{I}}{\Gamma \vdash \mathsf{rec}^{\mathcal{I}} (b_v \mid b_{\mathsf{app}} \mid b_{\mathtt{lam}})\ \Psi\ t : \{\Psi/\psi,\ t/y\}\tau}$$

Branches where $\quad \mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (y : \lceil \psi \vdash \mathsf{tm} \rceil) \Rightarrow \tau$

$$\frac{\Gamma, \psi : \mathsf{tm\_ctx}, p : \lceil \psi \vdash_{\#} \mathsf{tm} \rceil \vdash t_v : \{p/y\}\tau}{\Gamma \vdash (\psi, p \Rightarrow t_v) : \mathcal{I}}$$

$$\frac{\begin{array}{l}\Gamma, \psi : \mathsf{tm\_ctx}, m : \lceil \psi \vdash \mathsf{tm} \rceil, n : \lceil \psi \vdash \mathsf{tm} \rceil \\ \quad f_m : \{m/y\}\tau, f_n : \{n/y\}\tau \qquad \vdash \quad t_{\mathsf{app}} : \{\lceil \psi \vdash \mathsf{app}\ \lfloor m \rfloor_{\mathsf{id}}\ \lfloor n \rfloor_{\mathsf{id}} \rceil / y\}\tau\end{array}}{\Gamma \vdash (\psi, m, n, f_n, f_m \Rightarrow t_{\mathsf{app}}) : \mathcal{I}}$$

$$\frac{\begin{array}{l}\Gamma, \phi : \mathsf{tm\_ctx}, m : \lceil \phi, x : \mathsf{tm} \vdash \mathsf{tm} \rceil, \\ \quad f_m : \{(\phi, x : \mathsf{tm})/\psi, m/y\}\tau \qquad \vdash \quad t_{\mathtt{lam}} : \{\phi/\psi, \lceil \phi \vdash \mathtt{lam}\ \lambda x. \lfloor m \rfloor_{\mathsf{id}} \rceil / y\}\tau\end{array}}{\Gamma \vdash \psi, m, f_m \Rightarrow t_{\mathtt{lam}} : \mathcal{I}}$$

Fig. 10. Typing Rules for Recursors

Second statement:If $\mathcal{D} :: \Gamma; \Psi \vdash \mathcal{J}_{\mathsf{LF}}$ then $C ::\ \Gamma \vdash \Psi : \mathsf{ctx}$ and $C < \mathcal{D}$.

Case. $\mathcal{D} = \dfrac{\Gamma \vdash \Psi : \mathsf{ctx} \qquad \mathsf{a}{:}K \in \Sigma}{\Gamma; \Psi \vdash \mathsf{a} : K}$

$C :: \Gamma \vdash \Psi : \mathsf{ctx}$ and $C < \mathcal{D}$ by assumption

Case. $\mathcal{D} = \dfrac{\Gamma; \Psi, x{:}A \vdash M : B}{\Gamma; \Psi \vdash \lambda x.M : \Pi x{:}A.B}$

$C' :: \Gamma \vdash \Psi, x{:}A : \mathsf{ctx}$ and $C' < \mathcal{D}$ by IH

$C :: \Gamma \vdash \Psi : \mathsf{ctx}$ and $C < \mathcal{D}$ by inversion on well-formedness rules for LF contexts

□

Lemma 3.2 (LF Weakening). *Let* $\overrightarrow{y{:}B} = y_1{:}B_1, \ldots, y_k{:}B_k$.
If $\Gamma; \Psi, \overrightarrow{y{:}B} \vdash \mathcal{J}_{\mathsf{LF}}$ *and* $\Gamma \vdash (\Psi, x{:}A, \overrightarrow{y{:}B}) : \mathsf{ctx}$ *then* $\Gamma; \Psi, x{:}A, \overrightarrow{y{:}B} \vdash \mathcal{J}_{\mathsf{LF}}$.

Proof. By induction on the first derivation.

Reduction and Expansions for Computations

$$\frac{\Gamma \vdash \mathsf{fn}\ y \Rightarrow t : (y{:}\check{\tau}_1) \Rightarrow \tau_2 \quad \Gamma \vdash s : \check{\tau}_1}{\Gamma \vdash (\mathsf{fn}\ y \Rightarrow t)\ s \equiv \{s/y\}t : \{s/y\}\tau_2} \qquad \frac{\Gamma \vdash t : \lceil \Psi \vdash A \rceil}{\Gamma \vdash \lceil \hat{\Psi} \vdash \lfloor t \rfloor_{\mathsf{wk}_{\hat{\Psi}}} \rceil \equiv t : \lceil \Psi \vdash A \rceil}$$

let $\mathcal{B} = (\psi, p \Rightarrow t_p \mid \psi, m, n, f_m, f_n \Rightarrow t_{\mathsf{app}} \mid \psi, m, f_m \Rightarrow t_{\mathtt{lam}}\ )$
and $\mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (y : \lceil \psi \vdash \mathsf{tm} \rceil) \Rightarrow \tau$

$$\frac{\Gamma \vdash \Psi : \mathsf{tm\_ctx} \qquad \Gamma; \Psi, x{:}\mathsf{tm} \vdash M : \mathsf{tm} \qquad \Gamma \vdash \mathcal{I} : u}{\Gamma \vdash \mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ \lceil \hat{\Psi} \vdash \mathtt{lam}\ \lambda x.M \rceil \equiv \{\theta\}t_{\mathtt{lam}}\ : \{\Psi/\psi, \lceil \hat{\Psi} \vdash \mathtt{lam}\ \lambda x.M \rceil / y\}\tau}$$

where $\theta = \Psi/\psi,\ \lceil \hat{\Psi}, x \vdash M \rceil / m,\ \mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ (\Psi, x{:}\mathsf{tm})\ \lceil \hat{\Psi}, x \vdash M \rceil / f$

$$\frac{\Gamma \vdash \Psi : \mathsf{tm\_ctx} \qquad \Gamma; \Psi \vdash M : \mathsf{tm} \qquad \Gamma; \Psi \vdash N : \mathsf{tm} \qquad \Gamma \vdash \mathcal{I} : u}{\Gamma \vdash \mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi \lceil \hat{\Psi} \vdash \mathsf{app}\ M\ N \rceil \equiv \{\theta\}t_{\mathsf{app}}\ : \{\Psi/\psi, \lceil \hat{\Psi} \vdash \mathsf{app}\ M\ N \rceil / y\}\tau}$$

where $\theta = \Psi/\psi,\ \lceil \hat{\Psi} \vdash M \rceil / m,\ \lceil \hat{\Psi} \vdash N \rceil / n,\ \mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ \lceil \hat{\Psi} \vdash M \rceil / f_m,\ \mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ \lceil \hat{\Psi} \vdash N \rceil / f_n$

$$\frac{x{:}\mathsf{tm} \in \Psi \quad \Gamma \vdash \Psi : \mathsf{tm\_ctx} \qquad \Gamma \vdash \mathcal{I} : u}{\Gamma \vdash \mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ \lceil \hat{\Psi} \vdash x \rceil \equiv \{\Psi/\psi,\ \lceil \hat{\Psi} \vdash x \rceil / p\}t_p : \{\Psi/\psi, \lceil \Psi \vdash x \rceil / y\}\tau}$$

Fig. 11. Definitional Equality for Computations

Case. $\dfrac{\Gamma \vdash \Psi, \overrightarrow{y{:}B} : \mathsf{ctx} \qquad \mathsf{a}{:}K \in \Sigma}{\Gamma; \Psi, \overrightarrow{y{:}B} \vdash \mathsf{a} : K}$

$\Gamma \vdash \Psi, x{:}A, \overrightarrow{y{:}B} : \mathsf{ctx}$ — by assumption
$\Gamma; \Psi, x : A, \overrightarrow{y{:}B} \vdash \mathsf{a} : K$ — by rule

Case. $\dfrac{\Gamma; \Psi, \overrightarrow{y{:}B} \vdash A' : \mathsf{type} \qquad \Gamma; \Psi, \overrightarrow{y{:}B}, x'{:}A' \vdash B' : \mathsf{type}}{\Gamma; \Psi, \overrightarrow{y{:}B} \vdash \Pi x'{:}A'.B' : \mathsf{type}}$

$\Gamma; \Psi, x{:}A, \overrightarrow{y{:}B} \vdash A' : \mathsf{type}$ — by IH
$\Gamma \vdash \Psi, x{:}A, \overrightarrow{y{:}B} : \mathsf{ctx}$ — by assumption
$\Gamma \vdash \Psi, x{:}A, \overrightarrow{y{:}B}, x'{:}A' : \mathsf{ctx}$ — by rules for well-formed LF contexts
$\mathcal{D} :: \Gamma; \Psi, \overrightarrow{y{:}B}, x'{:}A' \vdash B' : \mathsf{type}$ — by premise
$\Gamma; \Psi, x{:}A, \overrightarrow{y{:}B}, x'{:}A' \vdash B' : \mathsf{type}$ — by IH
$\Gamma; \Psi, x{:}A, \overrightarrow{y{:}B} \vdash \Pi x'{:}A'.B' : \mathsf{type}$ — by rule

□

Lemma 3.3 (LF Variable Lookup). *Let* $\Gamma \vdash \Psi : \mathsf{ctx}$ *and* $\Psi(x) = A$.
If $\Gamma; \Phi \vdash \sigma : \Psi$ *then* $\Gamma; \Phi \vdash M : [\sigma/\hat{\Psi}]A$ *and* lookup $x\ [\sigma/\hat{\Psi}] = M$.

Proof. By induction $\Gamma; \Phi \vdash \sigma : \Psi$.

Case. $\dfrac{\Gamma \vdash \Psi, \overrightarrow{y{:}B} : \mathsf{ctx}}{\Gamma; \Psi, \overrightarrow{y{:}B} \vdash \mathsf{wk}_{\hat{\Psi}} : \Psi}$

$x \in \hat{\Psi}$ by assumption $\Psi(x) = A$
lookup x $[\mathsf{wk}_{\hat{\Psi}}/\hat{\Psi}] = x$ by definition of lookup
$(\Psi, \overrightarrow{y{:}B})(x) = A$ since $\Psi(x) = A$
$\Gamma; \Psi, \overrightarrow{y{:}B} \vdash x : A$ by typing rule

Case. $\dfrac{\Gamma; \Phi \vdash \sigma : \Psi' \qquad \Gamma; \Phi \vdash N : [\sigma/\hat{\Psi}']B}{\Gamma; \Phi \vdash \sigma, N : \Psi', y{:}B}$ where $\Psi'(x) = A$ and $x \neq y$

$\Gamma; \Phi \vdash M : [\sigma/\hat{\Psi}']A$ by IH
$\Gamma; \Phi \vdash M : [\sigma, N/\hat{\Psi}', y]A$ since $y \notin \mathsf{FV}(A)$

Case. $\dfrac{\Gamma; \Phi \vdash \sigma : \Psi' \qquad \Gamma; \Phi \vdash M : [\sigma/\hat{\Psi}']A}{\Gamma; \Phi \vdash \sigma, M : \Psi', x{:}A}$ where $(\Psi', x{:}A)(x) = A$

lookup x $[\sigma, M/\hat{\Psi}', x] = M$ by def. of lookup
$\Gamma; \Phi \vdash M : [\sigma/\hat{\Psi}']A$ by premise
$\Gamma; \Phi \vdash M : [\sigma, M/\hat{\Psi}', x]A$ since $x \notin \mathsf{FV}(A)$

□

Lemma 3.4 (LF Substitution). *If* $\Gamma; \Psi \vdash \mathcal{J}_{\mathsf{LF}}$ *and* $\Gamma; \Phi \vdash \sigma : \Psi$ *then* $\Gamma; \Phi \vdash [\sigma/\Psi]\mathcal{J}_{\mathsf{LF}}$.

Proof. By induction on the derivation on the first derivation using well-formedness of LF contexts (Lemma 3.1) and LF weakening (Lemma 3.2). In the LF variable case, we refer to Lemma 3.3. Most cases are straightforward; we only show a few cases, the others are similar.

Case. $\dfrac{\Gamma \vdash \Psi : \mathsf{ctx} \quad \Psi(x) = A}{\Gamma; \Psi \vdash x : A}$

$\Gamma; \Phi \vdash M : [\sigma/\Psi]A$ and lookup x $[\sigma/\hat{\Psi}] = M$ by Lemma 3.3
$\Gamma; \Phi \vdash [\sigma/\Psi]x : [\sigma/\Psi]A$ by subst. def.

Case. $\dfrac{\mathrm{a} : K \in \Sigma \qquad \Gamma \vdash \Psi : \mathsf{ctx}}{\Gamma; \Psi \vdash \mathrm{a} : K}$

$\Gamma \vdash \Phi : \mathsf{ctx}$ by Lemma 3.1
$[\sigma/\Psi](\mathrm{a} : K) \in \Sigma$ as K is closed and $[\sigma/\Psi]K = K$
$\Gamma; \Phi \vdash [\sigma/\Psi]\mathrm{a} : [\sigma/\Psi]K$ by rule and substitution def.

Case. $\dfrac{\Gamma; \Psi \vdash A : \mathsf{type} \qquad \Gamma; \Psi, x : A \vdash B : \mathsf{type}}{\Gamma; \Psi \vdash \Pi x{:}A.B : \mathsf{type}}$

$\Gamma; \Phi \vdash [\sigma/\Psi]A : \mathsf{type}$ by IH
$\Gamma \vdash \Phi : \mathsf{ctx}$ by the lemma 3.1 using $\Gamma; \Phi \vdash \sigma : \Psi$
$\Gamma \vdash \Phi, x : [\sigma/\Psi]A : \mathsf{ctx}$ by rule
$\Gamma; \Phi \vdash \sigma : \Psi$ by assumption
$\Gamma; \Phi, x : [\sigma/\Psi]A \vdash \sigma : \Psi$ by Lemma 3.2
$\Gamma; \Phi, x : [\sigma/\Psi]A \vdash x : ([\sigma/\hat{\Psi}]A)$ by rule

$\Gamma; \Phi, x : [\sigma/\Psi]A \vdash \sigma,\ x : \Psi, x : A$ by rule
$\Gamma; \Phi, x : [\sigma/\Psi]A \vdash [\sigma,\ x/\Psi, x]B : \mathsf{type}$ by IH
$\Gamma; \Phi \vdash \Pi x{:}[\sigma/\Psi]A.[\sigma, x/\Psi, x]B : \mathsf{type}$ by rule
$\Gamma; \Phi \vdash [\sigma/\Psi](\Pi x{:}A.B) : [\sigma/\Psi]\mathsf{type}$ by substitution def.

Case. $\dfrac{\Gamma \vdash t : [\Phi' \vdash A] \text{ or } \Gamma \vdash t : [\Phi' \vdash_{\#} A] \qquad \Gamma; \Psi \vdash \sigma' : \Phi'}{\Gamma; \Psi \vdash \lfloor t \rfloor_{\sigma'} : [\sigma'/\Phi']A}$

$\Gamma; \Phi \vdash [\sigma/\Psi]\sigma' : \Phi'$ IH
$\Gamma; \Phi \vdash \lfloor t \rfloor_{[\sigma/\Psi]\sigma'} : [[\sigma/\Psi]\sigma'/\Phi']A$ by rule
$\Gamma; \Phi \vdash [\sigma/\Psi](\lfloor t \rfloor_{\sigma'}) : [\sigma/\Psi]([\sigma'/\Phi']A)$ by substitution def.

Case. $\dfrac{\Gamma; \Psi \vdash M : \Pi x{:}A.B \qquad \Gamma; \Psi \vdash N : A}{\Gamma; \Psi \vdash M\ N : [N/x]B}$

$\Gamma; \Phi \vdash [\sigma/\Psi]M : [\sigma/\Psi](\Pi x{:}A.B)$ by IH
$\Gamma; \Phi \vdash [\sigma/\Psi]M : \Pi x{:}[\sigma/\Psi]A.([\sigma,\ x/\Psi, x]B)$ by substitution def.
$\Gamma; \Phi \vdash [\sigma/\Psi]N : [\sigma/\Psi]A$ by IH
$\Gamma; \Phi \vdash ([\sigma/\Psi]M)\ ([\sigma/\Psi]N) : [[\sigma/\Psi]N/x]([\sigma,\ x/\Psi, x]\ B)$ by rule
$\Gamma; \Phi \vdash [\sigma/\Psi](M\ N) : [\sigma/\Psi]([N/x]B)$ by definition and composition of substitution

Case. $\dfrac{\Gamma \vdash \Psi, \overrightarrow{x{:}A} : \mathsf{ctx}}{\Gamma; \Psi, \overrightarrow{x : A} \vdash \mathsf{wk}_{\hat{\Psi}} : \Psi}$

$\Gamma; \Phi \vdash \sigma : \Psi, \overrightarrow{x{:}A}$ by assumption
$\Gamma; \Phi \vdash \sigma' : \Psi$ by inversion where $\sigma = \sigma', M_n, \ldots M_1$
$\Gamma; \Phi \vdash [\sigma/\Psi, \vec{x}](\mathsf{wk}_{\hat{\Psi}}) : \Psi$ using the fact that $\sigma' = \mathsf{trunc}_{\Psi}(\sigma/\hat{\Psi}, \vec{x}) = [\sigma/\Psi, \vec{x}](\mathsf{wk}_{\hat{\Psi}})$

Case. $\dfrac{\Gamma; \Psi \vdash \sigma' : \Phi' \qquad \Gamma; \Psi \vdash M : [\sigma'/\Phi']A}{\Gamma; \Psi \vdash \sigma', M : \Phi', x : A}$

$\Gamma; \Phi \vdash [\sigma/\Psi]\sigma' : \Phi'$ by IH
$\Gamma; \Phi \vdash [\sigma/\Psi]M : [\sigma/\Psi]([\sigma'/\Phi']A$ by IH
$\Gamma; \Phi \vdash [\sigma/\Psi]M : [[\sigma/\Psi]\sigma'/\Phi']A$ by substitution def.
$\Gamma; \Phi \vdash [\sigma/\Psi]\sigma', [\sigma/\Psi]M : \Phi', x : A$ by rule
$\Gamma; \Phi \vdash [\sigma/\Psi](\sigma', M) : \Phi', x : A$ by substitution def.

□

We omit here the substitution lemma for the restricted LF typing judgments $\Gamma; \Psi \vdash_{\#} M : A$ and $\Gamma; \Phi \vdash_{\#} \sigma : \Psi$. However, it is worth noting that when we apply an LF substitution σ where $\Gamma; \Phi \vdash \sigma : \Psi$ to M where $\Gamma; \Psi \vdash_{\#} M : A$ we are not guaranteed to obtain a variable and hence we can only conclude $\Gamma; \Phi \vdash [\sigma/\hat{\Psi}]M : [\sigma/\hat{\Psi}]A$. We can only guarantee that we remain in $\vdash_{\#}$ if the LF substitution is a variable substitution.

Lemma 3.5 (LF Context Conversion). *Assume* $\Gamma \vdash \Psi, x{:}A : \mathsf{ctx}$ *and* $\Gamma; \Psi \vdash B : \mathsf{type}$.
If $\Gamma; \Psi, x{:}A \vdash \mathcal{J}_{\mathsf{LF}}$ *and* $\Gamma; \Psi \vdash A \equiv B : \mathsf{type}$ *then* $\Gamma; \Psi, x{:}B \vdash \mathcal{J}_{\mathsf{LF}}$.

Proof. Proof using LF Substitution (Lemma 3.4).

$\Gamma \vdash \Psi, x{:}A : \mathsf{ctx}$ by assumption
$\Gamma \vdash \Psi : \mathsf{ctx}$ by inversion
$\Gamma \vdash \Psi, x{:}B : \mathsf{ctx}$ by rule
$\Gamma; \Psi, x{:}B \vdash \mathsf{wk}_{\hat{\Psi}} : \Psi$ by rule
$\Gamma; \Psi, x{:}B \vdash x : B$ by rule
$\Gamma; \Psi \vdash B \equiv A : \mathsf{type}$ by symmetry
$\Gamma; \Psi, x{:}B \vdash x : A$ by rule
$\Gamma; \Psi, x{:}B \vdash x : [\mathsf{wk}_{\hat{\Psi}}/\hat{\Psi}]A$ as $[\mathsf{wk}_{\hat{\Psi}}/\hat{\Psi}]A = A$
$\Gamma; \Psi, x{:}B \vdash \mathsf{wk}_{\hat{\Psi}}, x : \Psi, x{:}A$ by rule
$\Gamma; \Psi, x{:}B \vdash \mathcal{J}$ by Lemma 3.4 and $[\mathsf{wk}_{\hat{\Psi}}, x/\hat{\Psi}, x]\mathcal{J} = \mathcal{J}$

□

Lemma 3.6 (Functionality of LF Typing).
Let $\Gamma; \Psi \vdash \sigma_1 : \Phi$ *and* $\Gamma; \Psi \vdash \sigma_2 : \Phi$, *and* $\Gamma; \Psi \vdash \sigma_1 \equiv \sigma_2 : \Phi$.
(1) If $\Phi = \Phi_i, x_i{:}A, \overrightarrow{y{:}A}$ *and* $\Gamma; \Phi \vdash x_i : A$ *then* $\Gamma; \Psi \vdash [\sigma_1/\hat{\Phi}](x_i) \equiv [\sigma_2/\hat{\Phi}](x_i) : [\sigma_1/\hat{\Phi}](A)$.
(2) If $\Gamma; \Phi \vdash \sigma : \Phi'$ *then* $\Gamma; \Psi \vdash [\sigma_1/\hat{\Phi}]\sigma \equiv [\sigma_2/\hat{\Phi}]\sigma : \Phi'$.
(3) If $\Gamma; \Phi \vdash M : A$ *then* $\Gamma; \Psi \vdash [\sigma_1/\hat{\Phi}]M \equiv [\sigma_2/\hat{\Phi}]M : [\sigma_1/\hat{\Phi}]A$.
(4) If $\Gamma; \Phi \vdash A : \mathsf{type}$ *then* $\Gamma; \Psi \vdash [\sigma_1/\hat{\Phi}]A \equiv [\sigma_2/\hat{\Phi}]A : \mathsf{type}$.

Proof. We prove these statements by induction on the typing derivation $\Gamma; \Phi \vdash M : A$ (resp. $\Gamma; \Phi \vdash \sigma : \Phi'$and $\Gamma; \Phi \vdash A : \mathsf{type}$) and followed by another inner induction on $\Gamma; \Psi \vdash \sigma_1 \equiv \sigma_2 : \Phi$ to prove (1).

We concentrate first on the variable case (1).

Case. $$\frac{\Gamma \vdash \Phi_0, x_0{:}A_0, \overrightarrow{y{:}B} : \mathsf{ctx}}{\Gamma; \Phi_0, x_0{:}A_0, \overrightarrow{y{:}B} \vdash \mathsf{wk}_{\hat{\Phi}, x} \equiv \mathsf{wk}_{\Phi_0}, x_0 : \Phi_0, x_0{:}A_0}$$

Let $x_i \in \hat{\Phi}_0$ and $\Phi_0 = \bullet, x_n, \ldots, x_1$ where $\bullet$ stands for either the empty context or a variable. Then lookup $x_i[\mathsf{wk}_{\bullet, x_n \ldots, x_1}/\hat{x}_n \ldots, x_1] = x_i$

Subcase. $x_i = x_0$
lookup x_i $[\mathsf{wk}_{\hat{\Phi}, x_0}/\hat{\Phi}_0, x_0] = x_0$ since $x_i{:}A_i \in (\Phi_0, x_0 : A_0)$
lookup x_i $[\mathsf{wk}_{\Phi_0}, x_0/\hat{\Phi}_0, x_0] = x_0$ by lookup

Subcase. $x \neq x_0$ and $x_i \in x_n, \ldots, x_1$
lookup x_i $[\mathsf{wk}_{\hat{\Phi}, x}/\hat{\Phi}_0, x_0] = x_i$ since $x_i{:}A_i \in (\Phi_0, x_0 : A_0)$
lookup x_i $[\mathsf{wk}_{\Phi_0}, x_0/\hat{\Phi}_0, x_0] =$ lookup x_i $[\mathsf{wk}_{\Phi_0}/\hat{\Phi}_0] = x_i$
since lookup $x_i[\mathsf{wk}_{\bullet, x_n \ldots, x_1}/\bullet, x_n \ldots, x_1] = x_i$
$\Gamma; \Phi_0, x_0{:}A_0, \overrightarrow{y{:}B} \vdash x_i \equiv x_i : A_i$ using $A_i = [\mathsf{wk}_{\bullet, x_n, \ldots, x_{i-1}}/\bullet, x_n, \ldots, x_{i-1}]A_i$

Case. $$\frac{\Gamma; \Psi \vdash \sigma = \sigma' : \Phi \quad \Gamma; \Psi \vdash M \equiv N : [\sigma/\hat{\Phi}]A}{\Gamma; \Psi \vdash \sigma, M \equiv \sigma', N : \Phi, y{:}A}$$

Subcase. $x = y$
lookup x $[\sigma, M/\hat{\Phi}, x{:}A] = M$ by def. of lookup

lookup x $[\sigma', N/\hat{\Phi}, x{:}A] = N$ by def. of lookup
$\Gamma; \Psi \vdash M \equiv N : [\sigma/\hat{\Phi}]A$ by premise
$\Gamma; \Psi \vdash M \equiv N : [\sigma, M/\hat{\Phi}, x]A$ since $[\sigma, M/\hat{\Phi}, x]A = [\sigma/\hat{\Phi}]A$

Case. $\dfrac{\Gamma; \Psi \vdash \sigma' \equiv \sigma : \Phi}{\Gamma; \Psi \vdash \sigma \equiv \sigma' : \Phi}$

$\Gamma; \Psi \vdash [\sigma'/\hat{\Phi}](x) \equiv [\sigma/\hat{\Phi}](x) : [\sigma'/\hat{\Phi}]A$ by IH
$\Gamma; \Psi \vdash [\sigma'/\hat{\Phi}]A \equiv [\sigma/\hat{\Phi}]A$ by IH
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}](x_i) \equiv [\sigma'/\hat{\Phi}](x_i) : [\sigma'/\hat{\Phi}](A)$ by type conversion

□

Lemma 3.7 (Equality Inversion). *If* $\Gamma; \Psi \vdash A \equiv \Pi x{:}B_1.B_2$: type *or* $\Gamma; \Psi \vdash \Pi x{:}B_1.B_2 \equiv A$: type *then* $A = \Pi x{:}A_1.A_2$ *for some* A_1 *and* A_2 *and* $\Gamma; \Psi \vdash A_1 \equiv B_1$: type *and* $\Gamma; \Psi, x{:}A_1 \vdash A_2 \equiv B_2$: type.

Proof. By induction on the definitional equality derivation. □

Lemma 3.8 (Injectivity of LF Pi-Types). *If* $\Gamma; \Psi \vdash \Pi x{:}A.B \equiv \Pi x{:}A'.B'$: type *then* $\Gamma; \Psi \vdash A \equiv A'$: type *and* $\Gamma; \Psi, x{:}A \vdash B \equiv B'$: type.

Proof. By equality inversion (Lemma 3.7). □

3.2 Elementary Properties of Computations

Theorem 3.9 (Well-Formedness of Computation Context).

(1) If $\mathcal{D} :: \vdash \Gamma, x{:}\check{\tau}, \Gamma'$ *then* $C :: \vdash \Gamma$ *and* C *is a sub-derivation of* $\mathcal{D}$*, i.e.* $C < \mathcal{D}$.
(2) If $\mathcal{D} :: \Gamma; \Psi \vdash \mathcal{J}_{\mathsf{LF}}$ *then* $C :: \vdash \Gamma$ *and* C *is a sub-derivation of* $\mathcal{D}$*, i.e.* $C < \mathcal{D}$.
(3) If $\mathcal{D} :: \Gamma \vdash \mathcal{J}_{\mathsf{comp}}$ *then* $C :: \vdash \Gamma$ *and* C *is a sub-derivation of* $\mathcal{D}$*, i.e.* $C < \mathcal{D}$.

Proof. (1) by induction on the structure of Γ'; (2) and (3) by mutual induction on $\mathcal{D}$.

First statement: If $\mathcal{D} :: \vdash \Gamma, x{:}\check{\tau}, \Gamma'$ then $C :: \vdash \Gamma$ and C is a sub-derivation of $\mathcal{D}$, i.e. $C < \mathcal{D}$.

Case. $\Gamma' = \cdot$

$\mathcal{D} :: \vdash \Gamma, x{:}\check{\tau}$ by assumption
$C :: \vdash \Gamma$ and $C < \mathcal{D}$ by inversion

Case. $\Gamma' = \Gamma'', y{:}\check{\tau}'$

$\mathcal{D} :: \vdash (\Gamma, x{:}\check{\tau}, \Gamma'', y{:}\check{\tau}')$ by assumption
$\mathcal{D}' :: \vdash \Gamma, x{:}\check{\tau}, \Gamma''$ and $\mathcal{D}' < \mathcal{D}$ by inversion
$\vdash \Gamma$ and $C < \mathcal{D}$ by IH

For the 2nd and 3rd statement we show a few cases; most of the cases are straightforward and follow either directly by applying the induction hypothesis or by the premises of a rule. We only show one case.

Case. $\mathcal{D} = \dfrac{\Gamma \vdash t : \tau}{\Gamma \vdash t \equiv t : \tau}$

$C :: \vdash \Gamma$ and $C < \mathcal{D}$ by IH

Case. $\mathcal{D} = \dfrac{\Gamma, y{:}\breve{\tau}_1 \vdash t \equiv s : \tau_2}{\Gamma \vdash \mathsf{fn}\ y \Rightarrow t \equiv \mathsf{fn}\ y \Rightarrow s : (y : \breve{\tau}_1) \Rightarrow \tau_2}$

$\mathcal{C}' :: \vdash \Gamma, y : \breve{\tau}_1$ and $\mathcal{C}' < \mathcal{D}$ by IH

$\mathcal{C} :: \vdash \Gamma$ and $\mathcal{C} < \mathcal{D}$ by well-formed context rule

□

Lemma 3.10 (Computation-level Weakening).

(1) If $\Gamma_1, \Gamma_2 \vdash \mathcal{J}_{\mathsf{comp}}$ and $\vdash \Gamma_1, y : \breve{\tau}, \Gamma_2$ then $\Gamma_1, y : \tau, \Gamma_2 \vdash \mathcal{J}_{\mathsf{comp}}$

(2) If $\Gamma_1, \Gamma_2; \Psi \vdash \mathcal{J}_{\mathsf{LF}}$ and $\vdash \Gamma_1, y : \breve{\tau}, \Gamma_2$ then $\Gamma_1, y : \breve{\tau}, \Gamma_2; \Psi \vdash \mathcal{J}_{\mathsf{LF}}$.

Proof. Proof by mutual induction exploiting Lemma 3.9.

Case. $\dfrac{\vdash \Gamma_1, \Gamma_2}{\Gamma_1, \Gamma_2 \vdash \cdot : \mathsf{ctx}}$

$\vdash \Gamma_1, y : \breve{\tau}, \Gamma_2$ by assumption

$\Gamma_1, y : \breve{\tau}, \Gamma_2 \vdash \cdot : \mathsf{ctx}$ by rule

Case. $\mathcal{D} = \dfrac{\Gamma_1, \Gamma_2, y' : \breve{\tau}_1 \vdash t : \tau_2 \qquad \Gamma_1, \Gamma_2 \vdash (y' : \breve{\tau}_1) \Rightarrow \tau_2 : u}{\Gamma_1, \Gamma_2 \vdash \mathsf{fn}\ y' \Rightarrow t : (y' : \breve{\tau}_1) \Rightarrow \tau_2}$

$\vdash \Gamma_1, y : \breve{\tau}, \Gamma_2$ by assumption

$\vdash \Gamma_1, \Gamma_2, y' : \breve{\tau}_1$ by Lemma 3.9

$\Gamma_1, \Gamma_2 \vdash \breve{\tau}_1 : u$ by inversion

$\Gamma_1, y : \breve{\tau}, \Gamma_2 \vdash \breve{\tau}_1 : u$ by IH

$\vdash \Gamma_1, y : \breve{\tau}, \Gamma_2, y' : \breve{\tau}_1$ by rule

$\Gamma_1, y : \breve{\tau}, \Gamma_2 \vdash (y' : \breve{\tau}_1) \Rightarrow \tau_2 : u$ by IH

$\Gamma_1, y : \tau, \Gamma_2, y' : \breve{\tau}_1 \vdash t : \tau_2$ by IH

$\Gamma_1, y : \tau, \Gamma_2 \vdash \mathsf{fn}\ y \Rightarrow' t : (y' : \breve{\tau}_1) \Rightarrow \tau_2$ by rule

Case. $\mathcal{D} = \dfrac{y' : \breve{\tau}' \in (\Gamma_1, \Gamma_2) \qquad \vdash (\Gamma_1, \Gamma_2)}{\Gamma_1, \Gamma_2 \vdash y' : \breve{\tau}'}$

$y' : \breve{\tau}' \in (\Gamma_1, y : \breve{\tau}, \Gamma_2)$ by since $y' : \breve{\tau}' \in (\Gamma_1, \Gamma_2)$

$\vdash \Gamma_1, y : \breve{\tau}, \Gamma_2$ by assumption

$\Gamma_1, y : \breve{\tau}, \Gamma_2 \vdash y' : \breve{\tau}'$ by rule

□

Lemma 3.11 (Computation-level Substitution).

(1) If $\vdash \Gamma, y : \breve{\tau}, \Gamma'$ and $\Gamma \vdash t : \breve{\tau}$ then $\vdash \Gamma, \{t/y\}\Gamma'$

(2) If $\Gamma, y : \breve{\tau}, \Gamma'; \Psi \vdash \mathcal{J}_{\mathsf{LF}}$ and $\Gamma \vdash t : \breve{\tau}$ then $\Gamma, \{t/y\}\Gamma'; \Psi \vdash \{t/y\}\mathcal{J}_{\mathsf{LF}}$.

(3) If $\Gamma, y : \breve{\tau}, \Gamma' \vdash \mathcal{J}_{\mathsf{comp}}$ and $\Gamma \vdash t : \breve{\tau}$ then $\Gamma, \{t/y\}\Gamma' \vdash \{t/y\}\mathcal{J}_{\mathsf{comp}}$.

Proof. By mutual induction on the first derivation exploiting Lemma 3.10.
We show here a few cases. Most cases are straightforward and only require us to apply the induction hypothesis.

Part 1.

Case. $\Gamma' = \cdot$

$\vdash \Gamma, y : \breve{\tau}$ by assumption

$\vdash \Gamma$ by inversion

Case. $\Gamma' = \Gamma_0, x{:}\check{\tau}_0$

$\vdash \Gamma, y{:}\check{\tau}, \Gamma_0$ and $\Gamma, y{:}\check{\tau}, \Gamma_0 \vdash \check{\tau}_0 : u$ — by inversion on assumption
$\vdash \Gamma, \{t/y\}\Gamma_0$ — by IH (part 1)
$\Gamma, \{t/y\}\Gamma_0 \vdash \{t/y\}\check{\tau}_0 : u$ — by IH (part 2)
$\vdash \Gamma, \{t/y\}\Gamma_0, \{t/y\}\check{\tau}_0$ — by rule
$\vdash \Gamma, \{t/y\}(\Gamma_0, x{:}\check{\tau}_0)$ — by subst. def.

Part 2.

Case. $\dfrac{y'{:}\check{\tau}' \in (\Gamma, y{:}\check{\tau}, \Gamma') \qquad \vdash \Gamma, y{:}\check{\tau}, \Gamma'}{\Gamma, y{:}\check{\tau}, \Gamma' \vdash y' : \check{\tau}'}$ where $y \neq y'$

Subcase: $y'{:}\check{\tau}' \in \Gamma$.

$\{t/y\}y' = y'$ — by subst. def.
$\{t/y\}\check{\tau}' = \check{\tau}'$ — by subst. def. and the fact that $y \notin \mathsf{FV}(\check{\tau}')$
$\vdash \Gamma, \{t/y\}\Gamma'$ — by IH (part 1)
$y{:}\check{\tau}' \in (\Gamma, \{t/y\}\Gamma')$ — by previous lines
$\Gamma, \{t/y\}\Gamma' \vdash y' : \check{\tau}'$ — by rule

Subcase: $y{:}\check{\tau}' \in \Gamma'$.

$\vdash \Gamma, \{t/y\}\Gamma'$ — by IH (part 1)
$y'{:}\{t/y\}\check{\tau}' \in \{t/y\}\Gamma'$ — by previous lines
$y{:}\{t/y\}\check{\tau}' \in (\Gamma, \{t/y\}\Gamma')$ — by previous lines
$\Gamma, \{t/y\}\Gamma' \vdash y' : \{t/y\}\check{\tau}'$ — by rule

Case. $\dfrac{y{:}\check{\tau} \in (\Gamma, y{:}\check{\tau}, \Gamma') \qquad \vdash \Gamma, y{:}\check{\tau}, \Gamma'}{\Gamma, y{:}\check{\tau}, \Gamma' \vdash y : \check{\tau}}$

$\Gamma \vdash t : \check{\tau}$ — by assumption
$\vdash \Gamma, \{t/y\}\Gamma'$ — by IH (part 1)
$\{t/y\}\check{\tau} = \check{\tau}$ — by subst. def. and the fact that $y \notin \mathsf{FV}(\tau)$
$\Gamma, \{t/y\}\Gamma' \vdash \{t/y\}y : \{t/y\}\check{\tau}$ — by Lemma 3.10

Case. $\dfrac{\Gamma, y{:}\check{\tau}, \Gamma', x{:}\check{\tau}_1 \vdash t' : \tau_2 \qquad \Gamma, y : \check{\tau}, \Gamma' \vdash (x : \check{\tau}_1) \Rightarrow \tau_2 : u}{\Gamma, y{:}\check{\tau}, \Gamma' \vdash \mathsf{fn}\ x \Rightarrow t' : (x{:}\check{\tau}_1) \Rightarrow \tau_2}$

$\Gamma, \{t/y\}(\Gamma', x{:}\check{\tau}_1) \vdash \{t/y\}t' : \{t/y\}\tau_2$ — by IH (part 2)
$\Gamma, \{t/y\}\Gamma', x{:}\{t/y\}\check{\tau}_1 \vdash \{t/y\}t' : \{t/y\}\tau_2$ — by subst. def.
$\Gamma, \{t/y\}\Gamma' \vdash \{t/y\}((x : \check{\tau}_1) \Rightarrow \tau_2) : \{t/y\}u$ — by IH (part 2)
$\Gamma, \{t/y\}\Gamma' \vdash (x : \{t/y\}\check{\tau}_1) \Rightarrow \{t/y, x/x\}\tau_2) : \{t/y\}u$ — by subst. def.
$\Gamma, \{t/y\}\Gamma' \vdash \{t/y\}(\mathsf{fn}\ x \Rightarrow t') : \{t/y\}((x{:}\check{\tau}_1) \Rightarrow \tau_2)$ — by rule and subst. def.

Case. $\dfrac{\Gamma, y{:}\check{\tau}, \Gamma' \vdash t' : (x : \check{\tau}_1) \Rightarrow \tau_2 \qquad \Gamma, y{:}\check{\tau}, \Gamma' \vdash s : \tau_1}{\Gamma, y{:}\tau, \Gamma' \vdash t'\ s : \{s/x\}\tau_2}$

$\Gamma, \{t/y\}\Gamma' \vdash \{t/y\}t' : (x : \{t/y\}\breve{\tau}_1) \Rightarrow \{t/y,\ x/x\}\tau_2$ by IH (part 2) and definition of substitution
$\Gamma, \{t/y\}\Gamma' \vdash \{t/y\}s : \{t/y\}\breve{\tau}_1$ by IH (part 2)
$\Gamma, \{t/y\}\Gamma' \vdash (\{t/y\}t')\ (\{t/y\}s) : \{\{t/y\}s/x\}(\{t/y, x/x\}\tau_2)$ by rule
$\Gamma, \{t/y\}\Gamma' \vdash \{t/y\}(t'\ s) : \{t/y\}(\{s/x\}\tau_2)$ by definition and composition rules of substitution

□

Last, we define simultaneous computation-level substitution using the judgment $\boxed{\Gamma' \vdash \theta : \Gamma}$. For simplicity, we overload the typing judgment simply writing $\Gamma \vdash t : \breve{\tau}$, although if $\breve{\tau} = \mathsf{tm\_ctx}$, t stands for a LF context.

$$\frac{\vdash \Gamma'}{\Gamma' \vdash \cdot : \cdot} \qquad \frac{\Gamma' \vdash \theta : \Gamma \quad \Gamma' \vdash t : \{\theta\}\breve{\tau}}{\Gamma' \vdash \theta, t/x : \Gamma, x : \breve{\tau}}$$

We distinguish between a substitution θ that provides instantiations for variables declared in the computation context Γ, and a renaming substitution ρ which maps variables in the computation context Γ to the same variables in the context Γ' where $\Gamma' = \Gamma, \overrightarrow{x{:}\breve{\tau}}$ and $\Gamma' \vdash \rho : \Gamma$. We write $\Gamma' \leq_\rho \Gamma$ for the latter. We note that the substitution properties also hold for renamings.

Lemma 3.12 (Well-Formed Contexts for Substitutions). *If* $\Gamma' \vdash \theta : \Gamma$ *then* $\vdash \Gamma'$.

Proof. By induction on the structure of the derivation of $\Gamma' \vdash \theta : \Gamma$.

Case. $\dfrac{\vdash \Gamma'}{\Gamma' \vdash \cdot : \cdot}$

$\vdash \Gamma'$ by premise

Case. $\dfrac{\Gamma' \vdash \theta : \Gamma \qquad \Gamma' \vdash t : \{\theta\}\breve{\tau}}{\Gamma' \vdash \theta, t/x : \Gamma, x : \breve{\tau}}$

$\vdash \Gamma'$ by IH

□

Lemma 3.13 (Weakening for computation-level substitutions). *Let y be a new name s.t. $y \notin dom(\Gamma')$. If $\Gamma' \vdash \theta : \Gamma$. and $\Gamma' \vdash \breve{\tau} : u$ then $\Gamma', y : \breve{\tau} \vdash \theta : \Gamma$.*

Proof. By induction on the first derivation using Lemma 3.10.

Case. $\dfrac{\vdash \Gamma'}{\Gamma' \vdash \cdot : \cdot}$

$\vdash \Gamma'$ by premise
$\Gamma' \vdash \breve{\tau} : u$ by assumption
$\vdash \Gamma', y : \breve{\tau}$ by rule
$\Gamma', y : \breve{\tau} \vdash \cdot : \cdot$ by rule

Case. $\dfrac{\Gamma' \vdash \theta : \Gamma \qquad \Gamma' \vdash t : \{\theta\}\breve{\tau}'}{\Gamma' \vdash \theta, t/x : \Gamma, x : \breve{\tau}'}$

$\Gamma', y : \breve{\tau} \vdash \theta : \Gamma$ by IH
$\Gamma', y : \breve{\tau} \vdash t : \{\theta\}\breve{\tau}'$ by Lemma 3.10
$\Gamma', y : \breve{\tau} \vdash \theta, t/x : \Gamma, x : \breve{\tau}'$ by rule

□

Corollary 3.14 (Identity Extension of Computation-level Substitution). *Let y be a new name s.t. $y \notin dom(\Gamma')$ and $y \notin dom(\Gamma)$. If $\Gamma' \vdash \theta : \Gamma$ and $\Gamma' \vdash \{\theta\}\check{\tau} : u$ then $\Gamma', y : \{\theta\}\check{\tau} \vdash \theta, y/y : \Gamma, y{:}\check{\tau}$.*

Proof.

$\Gamma', y : \{\theta\}\check{\tau} \vdash \theta : \Gamma$	by Lemma 3.13
$\vdash \Gamma', y : \{\theta\}\check{\tau}$	by Lemma 3.12
$\Gamma', y : \{\theta\}\check{\tau} \vdash y : \{\theta\}\check{\tau}$	by typing rule
$\Gamma', y : \{\theta\}\check{\tau} \vdash \theta, y/y : \Gamma, y : \check{\tau}$	by rule

□

Lemma 3.15 (Computation-level Simultaneous Substitution).

(1) If $\Gamma' \vdash \theta : \Gamma$ and $\Gamma; \Psi \vdash \mathcal{J}_{\mathsf{LF}}$ then $\Gamma'; \{\theta\}\Psi \vdash \{\theta\}\mathcal{J}_{\mathsf{LF}}$.
(2) If $\Gamma' \vdash \theta : \Gamma$ and $\Gamma \vdash \mathcal{J}_{\mathsf{comp}}$ then $\Gamma' \vdash \{\theta\}\mathcal{J}_{\mathsf{comp}}$.

Proof. By mutual induction on the second derivation using Lemma 3.9 and Lemma 3.13.

Case. $\dfrac{x{:}\check{\tau} \in \Gamma \qquad \vdash \Gamma}{\Gamma \vdash x : \check{\tau}}$

$\Gamma' \vdash \theta : \Gamma$	by assumption
$\Gamma' \vdash \theta_0, t/x : \Gamma_0, x{:}\check{\tau}$ and $\theta = \theta_0,\ t/x,\ \theta_1$ and $\Gamma = \Gamma_0, x{:}\check{\tau}, \Gamma_1$	by inversion
$\Gamma' \vdash t : \{\theta_0\}\check{\tau}$	by inversion
$\Gamma' \vdash t : \{\theta\}\check{\tau}$	since $\check{\tau}$ does not depend on the variable in $(x{:}\check{\tau}, \Gamma_1)$
$\Gamma' \vdash \{\theta\}x : \{\theta\}\check{\tau}$	since $\{\theta\}x = t$ and t does not depend on the variable in $(x{:}\check{\tau}, \Gamma_1)$

Case. $\dfrac{\Gamma \vdash t : (y : \check{\tau}_1) \Rightarrow \tau_2 \qquad \Gamma \vdash s : \check{\tau}_1}{\Gamma \vdash t\ s : \{s/y\}\tau_2}$

$\Gamma' \vdash \{\theta\}s : \{\theta\}\check{\tau}_1$	by IH
$\Gamma' \vdash \{\theta\}t : \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$	by IH
$\Gamma' \vdash \{\theta\}t : (y : \{\theta\}\check{\tau}_1) \Rightarrow \{\theta, y/y\}\tau_2$	by subst. definition
$\Gamma' \vdash (\{\theta\}t)\ (\{\theta\}s) : \{\{\theta\}s/y\}(\{\theta, y/y\}\tau_2)$	by rule
$\Gamma' \vdash \{\theta\}(t\ s) : \{\{\theta\}s/y\}(\{\theta, y/y\}\tau_2)$	by subst. definition
$\Gamma' \vdash \{\theta\}(t\ s) : \{\theta\}(\{s/y\}\tau_2)$	by compositionality of substitution

Case. $\dfrac{\Gamma, y : \check{\tau}_1 \vdash t : \tau_2}{\Gamma \vdash \mathsf{fn}\ y \Rightarrow t : (y : \check{\tau}_1) \Rightarrow \tau_2}$

$\Gamma' \vdash \theta : \Gamma$	by assumption
$\vdash \Gamma, y : \check{\tau}_1$	by Lemma 3.9
$\Gamma \vdash \check{\tau}_1 : u$	by inversion
$\Gamma' \vdash \{\theta\}\check{\tau}_1 : u$	by IH
$\Gamma', y : \{\theta\}\check{\tau}_1 \vdash \theta, y/y : \Gamma, y : \check{\tau}_1$	by Lemma 3.14
$\Gamma', y : \{\theta\}\check{\tau}_1 \vdash \{\theta, y/y\}t : \{\theta, y/y\}\tau_2$	by IH
$\Gamma' \vdash \mathsf{fn}\ y \Rightarrow \{\theta, y/y\}t : (y : \{\theta\}\check{\tau}_1) \Rightarrow \{\theta, y/y\}\tau_2$	by rule
$\Gamma' \vdash \{\theta\}(\mathsf{fn}\ y \Rightarrow t) : \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$	by subst. definition

Case. $\dfrac{\Gamma, \psi : \mathsf{tm\_ctx}, p : \lceil \psi \vdash_{\#} \mathsf{tm} \rceil \vdash t_v : \{p/y\}\tau}{\Gamma \vdash (\psi, p \Rightarrow t_v) : \mathcal{I}}$

where $\mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (y : \lceil \psi \vdash \mathsf{tm} \rceil) \Rightarrow \tau$

$\Gamma' \vdash \theta : \Gamma$ by assumption

$\vdash \Gamma, \psi : \mathsf{tm\_ctx}, p : \lceil \psi \vdash_{\#} \mathsf{tm} \rceil$ by Lemma 3.9

$\vdash \Gamma, \psi : \mathsf{tm\_ctx}$ by inversion

$\Gamma, \psi : \mathsf{tm\_ctx} \vdash \lceil \psi \vdash_{\#} \mathsf{tm} \rceil$ by inversion

$\Gamma' \vdash \mathsf{tm\_ctx} : u$ by typing rule

$\Gamma' \vdash \{\theta\}\mathsf{tm\_ctx} : u$ by subst. def. since $\{\theta\}\mathsf{tm\_ctx} = \mathsf{tm\_ctx}$

$\Gamma', \psi : \mathsf{tm\_ctx} \vdash \theta, \psi/\psi : \Gamma, \psi : \mathsf{tm\_ctx}$ by Lemma 3.14

$\Gamma', \psi : \mathsf{tm\_ctx} \vdash \lceil \psi \vdash_{\#} \mathsf{tm} \rceil : u$ by typing rules

let $\theta_v = \theta, \psi/\psi, p/p$

$\Gamma', \psi : \mathsf{tm\_ctx}, p : \lceil \psi \vdash_{\#} \mathsf{tm} \rceil \vdash \theta_v : \Gamma, \psi : \mathsf{tm\_ctx}, p : \lceil \psi \vdash_{\#} \mathsf{tm} \rceil$ by Lemma 3.14

$\Gamma', \psi : \mathsf{tm\_ctx}, p : \lceil \psi \vdash_{\#} \mathsf{tm} \rceil \vdash \{\theta_v\}t_v : \{p/y\}\{\theta_v\}\tau$ by IH and by definition of substitution

$\Gamma' \vdash \{\theta\}(\psi, p \Rightarrow t_v) : \{\theta\}\mathcal{I}$ by rule and by definition of substitution

□

Next, we show that we can always extend a renaming substitution.

Lemma 3.16 (Weakening of Renaming Substitutions). *Let y be a new name s.t. $y \notin dom(\Gamma')$. If $\Gamma' \leq_\rho \Gamma$ and $\Gamma' \vdash \tau : u$ then $\Gamma', y : \tau \leq_\rho \Gamma$.*

Proof. Follows from Lemma 3.13. □

Corollary 3.17 (Identity Extension of Renaming Computation-level Substitution). *Let y be a new name s.t. $y \notin dom(\Gamma')$ and $y \notin dom(\Gamma)$. If $\Gamma' \leq_\rho \Gamma$ and $\Gamma' \vdash \{\rho\}\check{\tau} : u$ then $\Gamma', y{:}\{\rho\}\check{\tau}, \Gamma' \leq_{\rho, y/y} \Gamma, y{:}\check{\tau}$.*

Proof.

$\Gamma', y : \{\rho\}\check{\tau} \leq_\rho : \Gamma$ by Lemma 3.16

$\Gamma', y : \{\rho\}\check{\tau} \leq_{\rho, y/y} : \Gamma, y : \check{\tau}$ by rule

□

Lemma 3.18 (Computation-level Renaming Lemma).

(1) If $\Gamma' \leq_\rho \Gamma$ and $\Gamma; \Psi \vdash \mathcal{J}_{\mathsf{LF}}$ then $\Gamma'; \{\rho\}\Psi \vdash \{\rho\}\mathcal{J}_{\mathsf{LF}}$.
(2) If $\Gamma' \leq_\rho \Gamma$ and $\Gamma \vdash \mathcal{J}_{\mathsf{comp}}$ then $\Gamma' \vdash \{\rho\}\mathcal{J}_{\mathsf{comp}}$.

Proof. By induction on the second derivation using Lemma 3.9 We show a few cases.

Case. $\mathcal{D} = \dfrac{x{:}\check{\tau} \in \Gamma \qquad \vdash \Gamma}{\Gamma \vdash x : \check{\tau}}$

$\Gamma' \leq_\rho \Gamma$ by assumption

$\Gamma' \leq_{\rho_0, x/x} \Gamma_0, x{:}\check{\tau}$ and $\rho = \rho_0, x/x, \rho_1$ and $\Gamma = \Gamma_0, x{:}\check{\tau}, \Gamma_1$ by inversion

$\Gamma' \vdash x : \{\rho_0\}\check{\tau}$ by inversion

$\Gamma' \vdash x : \{\rho\}\check{\tau}$ since τ does not depend on the variable in $(x{:}\check{\tau}, \Gamma_1)$

$\Gamma' \vdash \{\rho\}x : \{\rho\}\check{\tau}$ since $\{\rho\}x = y$

Case. $\mathcal{D} = \dfrac{\Gamma \vdash t : (y : \check{\tau}_1) \Rightarrow \tau_2 \qquad \Gamma \vdash s : \check{\tau}_1}{\Gamma \vdash t\ s : \{s/y\}\tau_2}$

$\Gamma' \vdash \{\rho\}s : \{\rho\}\check{\tau}_1$ by IH

$\Gamma' \vdash \{\rho\}t : \{\rho\}((y : \check{\tau}_1) \Rightarrow \tau_2)$ by IH

$\Gamma' \vdash \{\rho\}t : (y : \{\rho\}\check{\tau}_1) \Rightarrow \{\rho, y/y\}\tau_2$ by subst. definition

$\Gamma' \vdash (\{\rho\}t)\ (\{\rho\}s) : \{\{\rho\}s/y\}(\{\rho, y/y\}\tau_2)$ by rule

$\Gamma' \vdash \{\rho\}(t\ s) : \{\{\rho\}s/y\}(\{\rho, y/y\}\tau_2)$ by subst. definition

$\Gamma' \vdash \{\rho\}(t\ s) : \{\rho\}(\{s/y\}\tau_2)$ by compositionality of substitution

Case. $\mathcal{D} = \dfrac{\Gamma, y : \check{\tau}_1 \vdash t : \tau_2}{\Gamma \vdash \mathsf{fn}\ y \Rightarrow t : (y : \check{\tau}_1) \Rightarrow \tau_2}$

$\Gamma' \leq_\rho \Gamma$ by assumption

$\mathcal{C} ::\ \ \vdash \Gamma, y : \check{\tau}_1$ and moreover $\mathcal{C}$ is smaller than $\mathcal{D}$ by Lemma 3.9

$\Gamma \vdash \check{\tau}_1 : u$ by inversion

$\Gamma' \vdash \{\rho\}\check{\tau}_1 : u$ by IH

$\Gamma', y : \{\rho\}\check{\tau}_1 \leq_{\rho, y/y} \Gamma, y : \check{\tau}_1$ by Lemma 3.16

$\Gamma', y : \{\rho\}\check{\tau}_1 \vdash \{\rho, y/y\}t : \{\rho, y/y\}\tau_2$ by IH

$\Gamma' \vdash \mathsf{fn}\ y \Rightarrow \{\rho, y/y\}t : (y : \{\rho\}\check{\tau}_1) \Rightarrow \{\rho, y/y\}\tau_2$ by rule

$\Gamma' \vdash \{\rho\}(\mathsf{fn}\ y \Rightarrow t) : \{\rho\}((y : \check{\tau}_1) \Rightarrow \tau_2)$ by subst. definition

Case. $\mathcal{D} = \Gamma \vdash \mathsf{rec}^{\mathcal{I}}\,(\psi, p \Rightarrow t_v \mid \psi, m, n, f_n, f_m \Rightarrow t_{\mathsf{app}} \mid \psi, m, f_m \Rightarrow t_{\mathtt{lam}}\ )\ t : \{[\Psi]/\psi, t/y\}\tau$

where $\mathcal{I} = (\psi : \lceil \mathsf{tm\_ctx} \rceil) \Rightarrow (m : \lceil \psi \vdash \mathsf{tm} \rceil) \Rightarrow \tau$.

$\Gamma \vdash t : \lceil \Psi \vdash \mathsf{tm} \rceil$

$\Gamma, \psi{:}\mathsf{tm\_ctx}, m{:}\lceil\ \psi, x{:}\mathsf{tm} \vdash \mathsf{tm} \rceil, f_m{:}\{\lceil\ \psi, x{:}\mathsf{tm} \rceil/\psi, m/y\}\tau \vdash t_{\mathtt{lam}}\ : \{\psi/\psi, \lceil\ \psi \vdash \mathtt{lam}\ \lambda x.\lfloor m \rfloor_{\mathsf{id}} \rceil/y\}\tau$

$\Gamma, \psi{:}\mathsf{tm\_ctx}, m{:}\lceil\ \psi \vdash \mathsf{tm} \rceil, n{:}\lceil\ \psi \vdash \mathsf{tm} \rceil, f_m{:}\{m/y\}\tau, f_n : \{n/y\}\tau \vdash t_{\mathsf{app}}{:}\{\lceil\ \psi \vdash \mathsf{app}\ \lfloor m \rfloor_{\mathsf{id}}\ \lfloor n \rfloor_{\mathsf{id}} \rceil/y\}\tau$

$\Gamma, \psi{:}\mathsf{tm\_ctx}, p{:}\lceil\ \psi \vdash_{\#} \mathsf{tm} \rceil \vdash t_v : \{p/y\}\tau$ by premise

$\Gamma' \vdash \{\rho\}t : \{\rho\}\lceil \Psi \vdash \mathsf{tm} \rceil$ by IH

$\Gamma' \vdash \{\rho\}t : \lceil \{\rho\}\Psi \vdash \mathsf{tm} \rceil$ by subst. definition

$\Gamma' \vdash \mathsf{tm\_ctx}$ by rule

$\Gamma', \psi : \mathsf{tm\_ctx} \vdash \psi : \mathsf{tm\_ctx}$ by rule

$\Gamma', \psi : \mathsf{tm\_ctx} \vdash \psi : \mathsf{ctx}$ by rule

$\Gamma', \psi : \mathsf{tm\_ctx}; \psi \vdash \mathsf{tm} : \mathsf{type}$ by rule

$\Gamma', \psi : \mathsf{tm\_ctx} \vdash \psi \vdash_{\#} \mathsf{tm} : \mathsf{type}$ by rule

$\Gamma', \psi : \mathsf{tm\_ctx} \vdash \lceil\ \psi \vdash_{\#} \mathsf{tm} \rceil : u$ by rule

$\Gamma', \psi : \mathsf{tm\_ctx}, p : \lceil\ \psi \vdash_{\#} \mathsf{tm} \rceil \leq_{(\rho, \psi/\psi, p/p)} \Gamma, \psi : \mathsf{tm\_ctx}, m : \lceil\ \psi, x{:}\mathsf{tm} \vdash \mathsf{tm} \rceil$ by Lemma 3.16

$\Gamma', \psi : \mathsf{tm\_ctx}, p : \lceil\ \psi \vdash_{\#} \mathsf{tm} \rceil \vdash \{\psi/\psi, p/p\}t_v : \{\rho, \psi/\psi, p/y\}\tau$ by IH (since τ does not depend on Γ')

$\Gamma' \vdash \mathsf{tm\_ctx}$ by rule

$\Gamma', \psi : \mathsf{tm\_ctx} \vdash \psi : \mathsf{tm\_ctx}$ by rule

$\Gamma', \psi : \mathsf{tm\_ctx} \vdash \psi : \mathsf{ctx}$ by rule

$\Gamma', \psi : \mathsf{tm\_ctx} \vdash \psi, x{:}\mathsf{tm} : \mathsf{ctx}$ by rule

$\Gamma', \psi : \mathsf{tm\_ctx}; \psi, x{:}\mathsf{tm} \vdash \mathsf{tm} : \mathsf{type}$ by rule

$\Gamma', \psi : \mathsf{tm\_ctx} \vdash \lceil\ \psi, x{:}\mathsf{tm} \vdash \mathsf{tm} \rceil : u$ by rule

$\Gamma', \psi : \mathsf{tm\_ctx}, m : \lceil\ \psi, x : \mathsf{tm} \vdash \mathsf{tm} \rceil \vdash \{\lceil \psi, x{:}\mathsf{tm} \rceil/\psi, m/y\}\tau : u$ by assumption and Lemma 3.9

$\Gamma', \psi{:}\mathsf{tm\_ctx}, m{:}\lceil\ \psi, x : \mathsf{tm} \vdash \mathsf{tm} \rceil \leq_{(\rho, \psi/\psi, m/m)} \Gamma, \psi{:}\mathsf{tm\_ctx}, m{:}\lceil \psi, x{:}\mathsf{tm} \vdash \mathsf{tm} \rceil$ by Lemma 3.16

$\Gamma', \psi : \mathsf{tm\_ctx}, m : \lceil\ \psi, x : \mathsf{tm} \vdash \mathsf{tm} \rceil \vdash \{\psi/\psi, m/m\}(\{\lceil\ \psi, x{:}\mathsf{tm} \rceil/\psi, m/y\}\tau) : u$

by IH (since τ does not depend of Γ')

$\Gamma', \psi{:}\mathsf{tm\_ctx}, m{:}\lceil\ \psi, x : \mathsf{tm} \vdash \mathsf{tm} \rceil, f_m{:}\{\lceil\ \psi, x{:}\mathsf{tm} \rceil/\psi, m/y\}\tau$

$\quad \leq_{(\rho, \psi/\psi, m/m, f_m/f_m)} \Gamma, \psi{:}\mathsf{tm\_ctx}, m{:}\lceil\ \psi, x : \mathsf{tm} \vdash \mathsf{tm} \rceil, f_m{:}\{\lceil\ \psi, x{:}\mathsf{tm} \rceil/\psi, m/y\}\tau$ by Lemma 3.16

$\Gamma', \psi : \mathsf{tm\_ctx}, m : \lceil \psi, x : \mathsf{tm} \vdash \mathsf{tm} \rceil, f_m : \{\lceil \psi, x{:}\mathsf{tm} \rceil/\psi, m/y\}\tau$

$$\vdash \{\rho, \psi/\psi, m/m, f_m/f_m\} t_{\mathtt{lam}} : \{\psi/\psi, \lceil \psi \vdash \mathtt{lam}\, \lambda x. \lfloor m \rfloor_{\mathsf{id}} \rceil / y\}\tau$$

by IH (since τ does not depend on Γ')

With a similar argument, we have :

$\Gamma', \psi : \mathsf{tm\_ctx}, m : \lceil \psi \vdash \mathsf{tm} \rceil, n : \lceil \lfloor \psi \rfloor_{\vdash} \mathsf{tm} \rceil, f_m : \{m/y\}\tau, f_n : \{n/y\}\tau$

$$\leq_{\rho, \psi/\psi, m/m, n/n} \Gamma, \psi : \mathsf{tm\_ctx}, m : \lceil \psi \vdash \mathsf{tm} \rceil, n : \lceil \psi \vdash \mathsf{tm} \rceil, f_m : \{m/y\}\tau, f_n : \{n/y\}\tau$$

So we have by IH:

$\Gamma', \psi{:}\mathsf{tm\_ctx}, m{:}\lceil \psi \vdash \mathsf{tm} \rceil, n{:}\lceil \psi \vdash \mathsf{tm} \rceil, f_m{:}\{m/y\}\tau, f_n{:}\{n/y\}\tau \vdash \{\rho\} t_{\mathsf{app}} : \{\lceil \psi \vdash \mathsf{app}\, \lfloor m \rfloor_{\mathsf{id}}\, \lfloor n \rfloor_{\mathsf{id}} \rceil / y\}\tau$

So we have :

$\Gamma' \vdash \{\rho\} \mathsf{rec}^{\mathcal{I}} (\psi, p \Rightarrow t_v \mid \psi, m, n, f_n, f_m \Rightarrow t_{\mathsf{app}} \mid \psi, m, f_m \Rightarrow t_{\mathtt{lam}}\ )\ t : \{\rho\}(\{\lceil \Psi \rceil/\psi, t/y\}\tau)$

□

4 WEAK HEAD REDUCTION

Before we define the operational semantics of Cocon using weak head reduction, we characterize weak head normal forms for both, (contextual) LF and computations (Fig. 12 and Fig. 13). They are mutually defined. Any computation-level term t that is unboxed within an LF object (i.e.$\lfloor t \rfloor_\sigma$) must be neutral and not further trigger any reductions. We leave the substitution σ that is associated with it untouched. LF substitutions are in whnf.

$\boxed{\mathsf{whnf}\ M}$: LF term M is in weak head normal; $\boxed{\mathsf{wne}\ M}$: LF term M is in weak head neutral

$$\frac{}{\mathsf{whnf}\ \lambda x.M} \quad \frac{\mathsf{wne}\ t}{\mathsf{whnf}\ \lfloor t \rfloor_\sigma} \quad \frac{\mathsf{wne}\ M}{\mathsf{whnf}\ M} \quad \frac{}{\mathsf{wne}\ (\mathtt{lam}\ M)} \quad \frac{}{\mathsf{wne}\ (\mathsf{app}\ M\ N)} \quad \frac{}{\mathsf{wne}\ x}$$

$\boxed{\mathsf{whnf}\ \sigma}$: LF substitution σ is in weak head normal form

$$\frac{}{\mathsf{whnf}\ (\sigma, M)} \quad \frac{}{\mathsf{whnf}\ \cdot} \qquad \frac{}{\mathsf{whnf}\ \mathsf{wk}_\psi}$$

$\boxed{\mathsf{whnf}\ A}$: LF type A is in weak head normal form

$$\frac{}{\mathsf{whnf}\ \mathsf{a}\ M_1 \ldots M_n} \qquad \frac{\mathsf{whnf}\ A \quad \mathsf{whnf}\ B}{\mathsf{whnf}\ \Pi x{:}A.B}$$

Fig. 12. Normal and Neutral LF Terms and LF Types

LF types are always considered to be in whnf, as computation may only produce a contextual LF term, but not a contextual LF type. LF contexts are in weak head normal form, as we do not allow the embedding of computations that compute a context. Similarly, the erased context, described by $\hat{\Psi}$, is in weak head normal form. Further, contextual objects C and types T are considered to be in weak head normal form.

Computation-level expressions are in weak head normal form, if they do not trigger any further computation-level reductions. We consider boxed objects, i.e. $\lceil C \rceil$, and boxed types, i.e. $\lceil T \rceil$, in whnf. The contextual object C will be further reduced when we use them and have to unbox them. A computation-level term t is neutral (i.e. wne t) if it cannot be reduced further because it contains free variables. Variables are neutral, applications $t\ s$ are neutral, if t is neutral, and $\mathsf{rec}^{\mathcal{I}} (\psi, p \Rightarrow t_v \mid \psi, m, n, f_m, f_n \Rightarrow t_{\mathsf{app}} \mid \psi, m, f_m \Rightarrow t_{\mathsf{lam}})\ \Psi\ t$ is neutral, if t is neutral. We note that weakening preserves weak head normal forms.

Lemma 4.1 (Weakening preserves head normal forms).
For LF Terms:

(1) If whnf M and $\Gamma' \leq_\rho \Gamma$ and $\Gamma; \Psi \vdash M : A$ then whnf $\{\rho\}M$.

$\boxed{\mathsf{whnf}\ t}$: Computation (type) t is in weak head normal form

$$\frac{}{\mathsf{whnf}\ \lceil T \rceil} \quad \frac{}{\mathsf{whnf}\ (y : \tau_1) \Rightarrow \tau_2} \quad \frac{}{\mathsf{whnf}\ u} \quad \frac{}{\mathsf{whnf}\ (\mathsf{fn}\ y \Rightarrow t)} \quad \frac{}{\mathsf{whnf}\ \lceil C \rceil} \quad \frac{\mathsf{wne}\ t}{\mathsf{whnf}\ t}$$

$$\frac{}{\mathsf{wne}\ y} \quad \frac{\mathsf{wne}\ t}{\mathsf{wne}\ (t\ s)} \quad \frac{\mathsf{wne}\ s}{\mathsf{wne}\ (\mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ (\Psi)\ s)} \quad \frac{r = \lfloor t \rfloor_\sigma \quad \mathsf{wne}\ t}{\mathsf{wne}\ (\mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ (\Psi)\ \lceil \hat{\Psi} \vdash r \rceil)}$$

Fig. 13. Normal and Neutral Computations

(2) If wne M and $\Gamma' \leq_\rho \Gamma$ and $\Gamma; \Psi \vdash M : A$ then wne $\{\rho\}M$.

For LF Substitutions

(1) If whnf σ and $\Gamma' \leq_\rho \Gamma$ and $\Gamma; \Psi \vdash \sigma : \Phi$ then whnf $\{\rho\}\sigma$.
(2) If wne σ and $\Gamma' \leq_\rho \Gamma$ and $\Gamma; \Psi \vdash \sigma : \Phi$ then wne $\{\rho\}\sigma$.

For Computations:

(1) If whnf t and $\Gamma' \leq_\rho \Gamma$ and $\Gamma \vdash t : \tau$ then whnf $\{\rho\}t$.
(2) If wne t and $\Gamma' \leq_\rho \Gamma$ and $\Gamma \vdash t : \tau$ then wne $\{\rho\}t$.

Proof. By induction on the first derivation. □

We define weak head reductions for LF in Fig. 14 and for computations in Fig. 13. If an LF term is not already in whnf form, we have two cases: either we encounter an LF application $M\ N$ and we may need to beta-reduce or M reduces to $\lfloor t \rfloor_\sigma$. If t is neutral, then we are done; otherwise t reduces to a contextual object $\lceil \hat{\Psi} \vdash M \rceil$, and we continue to reduce $[\sigma/\hat{\Psi}]M$.

$\boxed{M \searrow_{\mathsf{LF}} N}$: LF Term M weak head reduces to N s.t. whnf N

$$\frac{\mathsf{whnf}\ M}{M \searrow_{\mathsf{LF}} M} \quad \frac{M \searrow_{\mathsf{LF}} \lambda x.M' \quad [N/x]M' \searrow_{\mathsf{LF}} R}{M\ N \searrow_{\mathsf{LF}} R} \quad \frac{M \searrow_{\mathsf{LF}} R \quad \mathsf{wne}\ R}{M\ N \searrow_{\mathsf{LF}} R\ N}$$

$$\frac{t \searrow \lceil \hat{\Psi} \vdash M \rceil \quad [\sigma/\hat{\Psi}]M \searrow_{\mathsf{LF}} N}{\lfloor t \rfloor_\sigma \searrow_{\mathsf{LF}} N} \quad \frac{t \searrow n \quad \mathsf{wne}\ n}{\lfloor t \rfloor_\sigma \searrow_{\mathsf{LF}} \lfloor n \rfloor_\sigma}$$

$\boxed{\sigma \searrow_{\mathsf{LF}} \sigma'}$: LF Substitution σ weak head reduces to σ' s.t. whnf σ'

$$\frac{\mathsf{whnf}\ \sigma}{\sigma \searrow_{\mathsf{LF}} \sigma} \quad \frac{}{\mathsf{wk}_{\cdot} \searrow_{\mathsf{LF}} \cdot} \quad \frac{}{\mathsf{wk}_{(\hat{\Psi}, x)} \searrow_{\mathsf{LF}} \mathsf{wk}_{\hat{\Psi}}, x}$$

Fig. 14. Weak Head Reductions for LF Terms, LF Substitutions, LF Contexts, and LF Contextual Terms.

If a computation-level term t is not already in whnf form, we have either an application $t_1\ t_2$ or a recursor. For an application $t_1\ t_2$, we reduce t_1. If it reduces to a function and we continue to beta-reduce otherwise we build a neutral application. For the recursor $\mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t$ we also consider different cases: 1) if t reduces to a neutral term, then we cannot proceed; 2) if t reduces to $\lceil \hat{\Psi}' \vdash M \rceil$, and then proceed to further reduce M. If the result is $\lfloor t' \rfloor_\sigma$, where t is neutral, then we cannot proceed; if the result is N where N is neutral, then we proceed and choose the appropriate branch in $\mathcal{B}$. We note that weak head reduction for LF and computation is deterministic.

Lemma 4.2 (Determinacy of whnf reduction).

(1) If $M \searrow_{\mathsf{LF}} N_1$ and $M \searrow_{\mathsf{LF}} N_2$ then $N_1 = N_2$.
(2) If $\sigma \searrow_{\mathsf{LF}} \sigma_1$ and $\sigma \searrow_{\mathsf{LF}} \sigma_2$ then $\sigma_1 = \sigma_2$.

$\boxed{t \searrow r}$: Computation-level term t weak head reduces to r s.t. whnf r

$$\frac{\text{whnf } t}{t \searrow t} \qquad \frac{t_1 \searrow \text{fn } y \Rightarrow t \quad \{t_2/y\}t \searrow v}{t_1\ t_2 \searrow v} \qquad \frac{t_1 \searrow w \quad \text{wne } w}{t_1\ t_2 \searrow w\ t_2}$$

$$\frac{t \searrow s \quad \text{wne } s}{\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t \searrow \text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ s} \qquad \frac{t \searrow \lceil \hat{\Psi} \vdash M \rceil \quad M \searrow_{\mathsf{LF}} \lfloor t' \rfloor_{\sigma} \quad \text{wne } t'}{\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t \searrow \text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ (\hat{\Psi} \vdash \lfloor t' \rfloor_{\sigma})}$$

$$\frac{t \searrow \lceil \hat{\Psi} \vdash M \rceil \quad M \searrow_{\mathsf{LF}} N \quad \text{wne } N \quad \mathcal{B} \ll (\Psi)\ (\hat{\Psi} \vdash N) \searrow v}{\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t \searrow v}$$

let $\mathcal{B} = (\psi, p \Rightarrow t_v \mid \psi, m, n, f_m, f_n \Rightarrow t_{\text{app}} \mid \psi, m, f_m \Rightarrow t_{\mathtt{lam}})$

$$\frac{\{\Psi/\psi,\ \lceil \hat{\Psi} \vdash M \rceil/m,\ \lceil \hat{\Psi} \vdash N \rceil/n,\ \text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ \lceil \hat{\Psi} \vdash M \rceil/f_m,\ \text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ \lceil \hat{\Psi} \vdash N \rceil/f_n\}t_{\text{app}} \searrow v}{\mathcal{B} \ll (\Psi)\ (\hat{\Psi} \vdash \text{app}\ M\ N) \searrow v}$$

$$\frac{\{\Psi/\psi,\ \lceil \hat{\Psi} \vdash x \rceil/p\}t_v \searrow v}{\mathcal{B} \ll (\Psi)\ (\hat{\Psi} \vdash x) \searrow v} \qquad \frac{\{\Psi/\psi,\ \lceil \hat{\Psi}, x \vdash M \rceil/m,\ \text{rec}^{\mathcal{I}}\ \mathcal{B}\ (\Psi, x{:}\text{tm})\ \lceil \hat{\Psi}, x \vdash M \rceil/f\}t_{\mathtt{lam}} \searrow v}{\mathcal{B} \ll (\Psi)\ (\hat{\Psi} \vdash \mathtt{lam}\ \lambda x.M) \searrow v}$$

Fig. 15. Weak Head Reductions for Computations

(3) If $t \searrow t_1$ and $t \searrow t_2$ then $t_1 = t_2$.

Proof. By inspection of the rules. ◻

Our semantic model for equivalence characterizes well-typed terms. To facilitate our further development we introduce the following notational abbreviations for well-typed weak head normal forms (see Def. 4.3) and show that whnf reductions are preserved under renamings and are stable under substitutions.

Definition 4.3 (Well-Typed Whnf).

$$\begin{array}{rcl}
\Gamma; \Psi \vdash M \searrow_{\mathsf{LF}} N : A & :\Leftrightarrow & \Gamma; \Psi \vdash M : A \text{ and } \Gamma; \Psi \vdash N : A \text{ and } \Gamma; \Psi \vdash M \equiv N : A \text{ and } M \searrow_{\mathsf{LF}} N \\
\Gamma; \Psi \vdash \sigma_1 \searrow_{\mathsf{LF}} \sigma_2 : \Phi & :\Leftrightarrow & \Gamma; \Psi \vdash \sigma_1 : \Phi \text{ and } \Gamma; \Psi \vdash \sigma_2 : \Phi \text{ and } \Gamma; \Psi \vdash \sigma_1 \equiv \sigma_2 : \Phi \text{ and } \sigma_1 \searrow_{\mathsf{LF}} \sigma_2 \\
\Gamma \vdash t \searrow t' : \tau & :\Leftrightarrow & \Gamma \vdash t : \tau \text{ and } \Gamma \vdash t' : \tau \text{ and } \Gamma \vdash t \equiv t' : \tau \text{ and } t \searrow t'
\end{array}$$

Lemma 4.4 (Weak Head Reductions preserved under Weakening).

(1) If $\Gamma; \Psi \vdash M \searrow_{\mathsf{LF}} N : A$ and $\Gamma' \leq_\rho \Gamma$ then $\Gamma'; \{\rho\}\Psi \vdash \{\rho\}M \searrow_{\mathsf{LF}} \{\rho\}N : \{\rho\}A$.
(2) If $\Gamma; \Psi \vdash \sigma \searrow_{\mathsf{LF}} \sigma' : \Phi$ and $\Gamma' \leq_\rho \Gamma$ then $\Gamma'; \{\rho\}\Psi \vdash \{\rho\}\sigma \searrow_{\mathsf{LF}} \{\rho\}\sigma' : \{\rho\}\Phi$.
(3) If $\Gamma \vdash t \searrow t' : \tau$ and $\Gamma' \leq_\rho \Gamma$ then $\Gamma' \vdash \{\rho\}t \searrow \{\rho\}t' : \{\rho\}\tau$.

Proof. By mutual induction on the first derivation using the computation-level substitution lemma 3.15, as renaming $\Gamma' \leq_\rho \Gamma$ are a special case of computation-level substitutions. ◻

Lemma 4.5 (LF Weak Head Reduction is stable under LF Substitutions). *Let $\Gamma; \Psi \vdash \sigma : \Phi$.*

(1) If $\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{\sigma_1} : A$ then $\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1} : [\sigma/\hat{\Phi}]A$.
(2) If $\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} \lambda x.N : \Pi x{:}A.B$ then $\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\lambda x.N) : [\sigma/\hat{\Phi}](\Pi x{:}A.B)$.
(3) If $\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} x : A$ and $\Gamma; \Psi \vdash \sigma(x) \searrow_{\mathsf{LF}} N : [\sigma/\hat{\Phi}]A$ then $\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} N : [\sigma/\hat{\Phi}]A$.
(4) If $\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} \text{app}\ M_1\ M_2 : \text{tm}$ then $\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\text{app}\ M_1\ M_2) : \text{tm}$.
(5) If $\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} \mathtt{lam}\ M_1 : \text{tm}$ then $\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\mathtt{lam}\ M_1) : \text{tm}$.
(6) If $\Gamma; \Phi \vdash \sigma_1 \searrow_{\mathsf{LF}} \sigma_2 : \Phi'$ then $\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]\sigma_1 \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}]\sigma_2 : \Phi'$.

Proof. By induction on $M \searrow_{\mathsf{LF}} M'$ relation that is part of the well-typed weak head reduction using Lemma 3.1 and 3.4.

For (1): $\boxed{\text{If } \Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{\sigma_1} : A \text{ then } \Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1} : [\sigma/\hat{\Phi}]A.}$

Case $M = \lfloor t_0 \rfloor_{\sigma_1}$ and $t_0 \searrow t_1$ and wne t_1

$\Gamma; \Phi \vdash M : A$	by assumption
$\Gamma; \Phi \vdash \lfloor t_1 \rfloor_{\sigma_1} : A$	by assumption
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M : [\sigma/\hat{\Phi}]A$	by LF subst. lemma 3.4
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}](\lfloor t_1 \rfloor_{\sigma_1}) : [\sigma/\hat{\Phi}]A$	by LF subst. lemma 3.4
$[\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1}$	since wne t_1 and $t_0 \searrow t_1$ and LF subst. prop.
$\Gamma; \Phi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1} : [\sigma/\hat{\Phi}]A$	by well-typed whnf (Def 4.3)

Case $M = \lfloor t_0 \rfloor_{\sigma_0}$ and $t_0 \searrow \lceil \hat{\Phi}' \vdash M' \rceil$, $[\sigma_0/\hat{\Phi}']M' \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{\sigma_1}$

$\Gamma; \Phi \vdash M : A$	by assumption
$\Gamma; \Phi \vdash \lfloor t_1 \rfloor_{\sigma_1} : A$	by assumption
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M : [\sigma/\hat{\Phi}]A$	by LF subst. lemma 3.4
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}](\lfloor t_1 \rfloor_{\sigma_1}) : [\sigma/\hat{\Phi}]A$	by LF subst. lemma 3.4
$[[\sigma/\hat{\Phi}]\sigma_0/\hat{\Phi}']M' \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1}$	by IH (and subst. prop)
$\lfloor t_0 \rfloor_{[\sigma/\hat{\Phi}]\sigma_0} \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1}$	by whnf
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}](\lfloor t_0 \rfloor_{\sigma_0}) \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1} : [\sigma/\hat{\Phi}]A$	by subst. prop. and well-typed whnf (Def 4.3)

Case $M = \lfloor t_1 \rfloor_{\sigma_1}$ and wne t_1

$\Gamma; \Phi \vdash M : A$	by assumption
$\Gamma; \Phi \vdash \lfloor t_1 \rfloor_{\sigma_1} : A$	by assumption
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M : [\sigma/\hat{\Phi}]A$	by LF subst. lemma 3.4
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}](\lfloor t_1 \rfloor_{\sigma_1}) : [\sigma/\hat{\Phi}]A$	by LF subst. lemma 3.4
$[\sigma/\hat{\Phi}](\lfloor t_1 \rfloor_{\sigma_1}) = \lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1}$	by subst. def.
wne $\lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1}$	since wne t_1
$[\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1}$	by whnf
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1} : [\sigma/\hat{\Phi}]A$	by well-typed whnf (Def 4.3)

Case $M = M_1\ M_2$ and $M \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{\sigma_1}$

$\Gamma; \Phi \vdash M : A$	by assumption
$\Gamma; \Phi \vdash \lfloor t_1 \rfloor_{\sigma_1} : A$	by assumption
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M : [\sigma/\hat{\Phi}]A$	by LF subst. lemma 3.4
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}](\lfloor t_1 \rfloor_{\sigma_1}) : [\sigma/\hat{\Phi}]A$	by LF subst. lemma 3.4
$M_1 \searrow_{\mathsf{LF}} \lambda x.M'$ and $[M_2/x]M' \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{\sigma_1}$	by inversion
$[\sigma/\hat{\Phi}][M_2/x]M' \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\lfloor t_1 \rfloor_{\sigma_1})$	by IH
$[\sigma, [\sigma/\hat{\Phi}]M_2/\hat{\Phi}, x]M' \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\lfloor t_1 \rfloor_{\sigma_1})$	by subst. def
$[\sigma/\hat{\Phi}]M_1 \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\lambda x.M')$	by IH
$[\sigma/\hat{\Phi}]M_1 \searrow_{\mathsf{LF}} \lambda x.[\sigma, x/\hat{\Phi}, x]M'$	by subst. def.
$[[\sigma/\hat{\Phi}]M_2/x](\sigma, x) = \sigma, [\sigma/\hat{\Phi}]M_2$	by subst. def.
$[\sigma/\hat{\Phi}]M_1\ [\sigma/\hat{\Phi}]M_2 \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\lfloor t_1 \rfloor_{\sigma_1})$	by whnf rules
$[\sigma/\hat{\Phi}](M_1\ M_2) \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1}$	by subst. def.

For (3): $\boxed{\text{If } \Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} x : A \text{ and } \Gamma; \Psi \vdash \sigma(x) \searrow_{\mathsf{LF}} N : [\sigma/\hat{\Phi}]A \text{ then } \Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} N : [\sigma/\hat{\Phi}]A.}$

Case $M = x$ where $x \in \Phi$ and whnf M

$\Gamma; \Phi \vdash x : A$ and $x : A \in \Phi$	since $\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} x : A$
$\Gamma; \Phi \vdash M : A$	since $\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} x : A$
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M : [\sigma/\hat{\Phi}]A$	LF subst. lemma 3.4
$\Gamma; \Psi \vdash N : [\sigma/\hat{\Phi}]A$	since $\Gamma; \Psi \vdash \sigma : \Phi$ and $x : A \in \Phi$
$[\sigma/\hat{\Phi}]M = [\sigma/\hat{\Phi}]x = \sigma(x)$	by subst. def.
$[\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} N$	since $\sigma(x) \searrow_{\mathsf{LF}} N$

Case $M = M_1\ M_2$ and $M \searrow_{\mathsf{LF}} x$

$\Gamma; \Phi \vdash x : A$ and $x : A \in \Phi$	since $\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} x : A$
$\Gamma; \Phi \vdash M : A$	since $\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} x : A$
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M : [\sigma/\hat{\Phi}]A$	LF subst. lemma 3.4
$M_1 \searrow_{\mathsf{LF}} \lambda x.M'$ and $[M_2/x]M' \searrow_{\mathsf{LF}} x$	by inversion
$[\sigma/\hat{\Phi}][M_2/x]M' \searrow_{\mathsf{LF}} N$	by IH using $\sigma(x) \searrow_{\mathsf{LF}} N$
$[\sigma, [\sigma/\hat{\Phi}]M_2/\hat{\Phi}, x]M' \searrow_{\mathsf{LF}} N$	by subst. def.
$[\sigma/\hat{\Phi}]M_1 \searrow_{\mathsf{LF}} \lambda x.[\sigma, x]M'$	by IH and LF subst. prop.
$[\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} N$	by whnf rules
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} N : [\sigma/\hat{\Phi}]A$	by well-typed whnf (Def 4.3)

Case $M = \lfloor t_1 \rfloor_{\sigma_1}$ and $M \searrow_{\mathsf{LF}} x$

$\Gamma; \Phi \vdash x : A$ and $x : A \in \Phi$	since $\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} x : A$
$\Gamma; \Phi \vdash M : A$	since $\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} x : A$
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M : [\sigma/\hat{\Phi}]A$	LF subst. lemma 3.4
$t_1 \searrow \lceil \hat{\Phi}' \vdash M' \rceil$	since $\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} x : \mathsf{tm}$
$\sigma_1 \searrow_{\mathsf{LF}} \sigma_2$ and $[\sigma_2/\hat{\Phi}']M' \searrow_{\mathsf{LF}} x$	since $\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} x : \mathsf{tm}$
$[\sigma/\hat{\Phi}]\sigma_1 \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}]\sigma_2$	by IH
$[\sigma/\hat{\Phi}]([\sigma_2/\hat{\Phi}']M') \searrow_{\mathsf{LF}} N$	by IH
$[[\sigma/\hat{\Phi}]\sigma_2/\hat{\Phi}']M' \searrow_{\mathsf{LF}} N$	by subst. prop.
$\lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1} \searrow_{\mathsf{LF}} N$	by whnf rules
$\Gamma; \Psi \vdash \lfloor t_1 \rfloor_{[\sigma/\hat{\Phi}]\sigma_1} \searrow_{\mathsf{LF}} N : \mathsf{tm}$	by well-typed whnf (Def 4.3)

For (6): $\boxed{\text{If } \Gamma; \Phi \vdash \sigma_1 \searrow_{\mathsf{LF}} \sigma_2 : \Phi' \text{ then } \Gamma; \Psi \vdash [\sigma/\hat{\Phi}]\sigma_1 \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}]\sigma_2 : \Phi'.}$

Case. whnf σ_1 and $\sigma_2 = \sigma_1$

whnf $[\sigma/\hat{\Phi}]\sigma_1$	by def. of whnf
$[\sigma/\hat{\Phi}]\sigma_1 \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}]\sigma_1$	by whnf
$\Gamma; \Phi \vdash \sigma_1 : \Phi'$ and $\Gamma; \Phi \vdash \sigma_2 : \Phi'$	by assumption
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]\sigma_1 : \Phi$ and $\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]\sigma_2 : \Phi$	by LF subst. lemma
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]\sigma_1 \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}]\sigma_1 : \Phi'$	by well-typed whnf (Def 4.3)

Case. $\sigma_1 = \mathsf{wk}.$

$\Gamma; \Phi \vdash \mathsf{wk}. : \cdot$	by assumption
$\Gamma \vdash \Phi : \mathsf{ctx}$ and $\Gamma \vdash \Psi : \mathsf{ctx}$	by well-formedness of LF context (Lemma 3.1)
$\Gamma; \Phi \vdash \cdot : \cdot$	by typing rule
$\cdot = [\sigma/\hat{\Phi}]\mathsf{wk}. = \mathsf{trunc}.(\sigma/\hat{\Phi})$	by subst. def.
$\Gamma; \Psi \vdash \cdot : \cdot$	by typing rule
$\mathsf{whnf}\ \cdot$	by whnf
$\cdot \searrow_{\mathsf{LF}} \cdot$	by whnf
$[\sigma/\hat{\Phi}]\mathsf{wk}. \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}]\cdot$	since $\cdot = [\sigma/\hat{\Phi}]\mathsf{wk}.$ and $[\sigma/\hat{\Phi}]\cdot = \cdot$
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]\mathsf{wk}. \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}]\cdot : \cdot$	by well-typed whnf (Def 4.3)

Case. $\sigma_1 = \mathsf{wk}_{\hat{\Phi}', x}$ where $\Phi = \Phi', x{:}A, \overrightarrow{x{:}A}$

$\sigma_2 = \mathsf{wk}_{\hat{\Phi}'}, x$	by assumption
$\Gamma; \Phi \vdash \mathsf{wk}_{\hat{\Phi}', x} : \Phi', x{:}A$ and $\Gamma; \Phi \vdash \mathsf{wk}_{\hat{\Phi}'} : \Phi'$	by assumption and typing
$\Gamma; \Psi \vdash \sigma : \Phi', x{:}A, \overrightarrow{x{:}A}$	by assumption

Sub-case : $\sigma = (\sigma', M, \vec{M})$

$\Gamma; \Psi \vdash (\sigma', M, \vec{M}) : \Phi', x{:}A, \overrightarrow{x{:}A}$	where $\sigma = (\sigma', M, \vec{M})$
$\Gamma; \Psi \vdash \sigma' : \Phi'$ and $\Gamma; \Psi \vdash M : [\sigma'/\hat{\Phi}']A$	by inversion
$[\sigma/\hat{\Phi}](\mathsf{wk}_{\hat{\Phi}', x}) = \mathsf{trunc}_{\hat{\Phi}', x}(\sigma/\Phi) = \sigma', M$	by definition
$[\sigma/\hat{\Phi}](\mathsf{wk}_{\hat{\Phi}'}) = \mathsf{trunc}_{\hat{\Phi}'}(\sigma/\Phi) = \sigma'$	by definition
$\Gamma; \Psi \vdash \sigma', M \searrow_{\mathsf{LF}} \sigma', M : \Phi', x{:}A$	since wne (σ', M)
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}](\mathsf{wk}_{\hat{\Phi}', x}) \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\mathsf{wk}_{\hat{\Phi}'}, x) : \Phi', x{:}A$	by well-typed whnf (Def 4.3)

Sub-case : $\sigma = \mathsf{wk}_{\hat{\Phi}', x, \vec{x}}$

$[\sigma/\hat{\Phi}](\mathsf{wk}_{\hat{\Phi}', x}) = \mathsf{trunc}_{\hat{\Phi}', x}(\sigma/\Phi) = \mathsf{trunc}_{\hat{\Phi}', x}(\mathsf{wk}_{\hat{\Phi}', x, \vec{x}}/\hat{\Phi}', x, \vec{x}) = \mathsf{wk}_{\hat{\Phi}', x}$	by definition
$[\sigma/\hat{\Phi}](\mathsf{wk}_{\hat{\Phi}'}, x) = \mathsf{trunc}_{\hat{\Phi}'}(\sigma/\Phi) = \mathsf{wk}_{\hat{\Phi}'}$	by definition
$[\sigma/\hat{\Phi}](x) = x$	since $\sigma = \mathsf{wk}_{\hat{\Phi}', x, \vec{x}}$
$\Gamma; \Phi', x{:}A, \overrightarrow{x : A} \vdash \mathsf{wk}_{\hat{\Phi}', x} \searrow_{\mathsf{LF}} \mathsf{wk}_{\hat{\Phi}'}, x : \Phi', x{:}A$	by $\searrow_{\mathsf{LF}}$ rule and by well-typed whnf (Def 4.3)

$\boxed{\text{For (4): If } \Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} \mathsf{app}\ M_1\ M_2 : \mathsf{tm} \text{ then } \Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\mathsf{app}\ M_1\ M_2) : \mathsf{tm}.}$

Case. $\dfrac{\mathsf{whnf}\ (\mathsf{app}\ M_1\ M_2)}{\mathsf{app}\ M_1\ M_2 \searrow_{\mathsf{LF}} \mathsf{app}\ M_1\ M_2}$

$\Gamma; \Phi \vdash \mathsf{app}\ M_1\ M_2 : \mathsf{tm}$	by assumption
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}](\mathsf{app}\ M_1\ M_2) : \mathsf{tm}$	by LF subst. lemma
$\Gamma; \Psi \vdash \mathsf{app}\ [\sigma/\hat{\Phi}](M_1)\ [\sigma/\hat{\Phi}](M_2) : \mathsf{tm}$	subst. prop
$\mathsf{whnf}\ (\mathsf{app}\ [\sigma/\hat{\Phi}](M_1)\ [\sigma/\hat{\Phi}](M_2))$	by whnf def.
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}](\mathsf{app}\ M_1\ M_2) \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\mathsf{app}\ M_1\ M_2) : \mathsf{tm}$	by subst. prop., $\searrow_{\mathsf{LF}}$ rule, and Def. 4.3

Case. $\dfrac{M \searrow_{\mathsf{LF}} \lambda x.M' \quad [N/x]M' \searrow_{\mathsf{LF}} \mathsf{app}\ M_1\ M_2}{M\ N \searrow_{\mathsf{LF}} \mathsf{app}\ M_1\ M_2}$

$\Gamma; \Phi \vdash \mathsf{app}\ M_1\ M_2 : \mathsf{tm}$ by assumption
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}](\mathsf{app}\ M_1\ M_2) : \mathsf{tm}$ by LF subst. lemma
$\Gamma; \Phi \vdash M\ N : \mathsf{tm}$ by assumption
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}](M\ N) : \mathsf{tm}$ by LF subst. lemma
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\lambda x.M') : \mathsf{tm}$ by IH
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]([N/x]M') \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\mathsf{app}\ M_1\ M_2) : \mathsf{tm}$ by IH
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}](M\ N) \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\mathsf{app}\ M_1\ M_2) : \mathsf{tm}$ with $[\sigma/\hat{\Phi}]([N/x]M') = [\sigma, [\sigma/\hat{\Phi}]N/\hat{\Phi}, x{:}\mathsf{tm}]$

Case. $\dfrac{t \searrow \lceil \hat{\Phi}' \vdash M \rceil \quad [\sigma'/\hat{\Phi}']M \searrow_{\mathsf{LF}} \mathsf{app}\ M_1\ M_2}{\lfloor t \rfloor_{\sigma'} \searrow_{\mathsf{LF}} \mathsf{app}\ M_1\ M_2}$

$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}][\sigma'/\hat{\Phi}']M \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\mathsf{app}\ M_1\ M_2) : \mathsf{tm}$ by IH
$\Gamma; \Psi \vdash \lfloor t \rfloor_{[\sigma/\hat{\Phi}]\sigma'} \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\mathsf{app}\ M_1\ M_2) : \mathsf{tm}$ by subst. prop., $\searrow_{\mathsf{LF}}$ rule, and Def. 4.3

The remaining cases for (2) and (5) are similar. □

5 KRIPKE-STYLE LOGICAL RELATION

We construct a Kripke-logical relation to prove weak head normalization. Our semantic definitions for computations follows closely Abel and Scherer [2012a] to accommodate type-level computation.

$$\dfrac{\begin{array}{llll}\Gamma; \Psi \vdash M \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{\sigma_1} : \mathsf{tm} & \mathsf{typeof}(\Gamma \vdash t_1) = \lceil \Phi_1 \vdash \mathsf{tm} \rceil & \Gamma \vdash \Phi_1 \equiv \Phi_2 : \mathsf{ctx} & \\ \Gamma; \Psi \vdash N \searrow_{\mathsf{LF}} \lfloor t_2 \rfloor_{\sigma_2} : \mathsf{tm} & \mathsf{typeof}(\Gamma \vdash t_2) = \lceil \Phi_2 \vdash \mathsf{tm} \rceil & \Gamma \vdash t_1 \equiv t_2 : \lceil \Phi_1 \vdash \mathsf{tm} \rceil & \Gamma; \Psi \Vdash \sigma_1 = \sigma_2 : \Phi_1\end{array}}{\Gamma; \Psi \Vdash M = N : \mathsf{tm}}$$

$$\dfrac{\begin{array}{l}\Gamma; \Psi \vdash M \searrow_{\mathsf{LF}} \mathtt{lam}\ M' : \mathsf{tm} \\ \Gamma; \Psi \vdash N \searrow_{\mathsf{LF}} \mathtt{lam}\ N' : \mathsf{tm} \quad \Gamma; \Psi, x{:}\mathsf{tm} \Vdash M'\ x = N'\ x : \mathsf{tm}\end{array}}{\Gamma; \Psi \Vdash M = N : \mathsf{tm}}$$

$$\dfrac{\begin{array}{l}\Gamma; \Psi \vdash M \searrow_{\mathsf{LF}} \mathsf{app}\ M_1\ M_2 : \mathsf{tm} \\ \Gamma; \Psi \vdash N \searrow_{\mathsf{LF}} \mathsf{app}\ N_1\ N_2 : \mathsf{tm} \quad \Gamma; \Psi \Vdash M_1 = N_1 : \mathsf{tm} \quad \Gamma; \Psi \Vdash M_2 = N_2 : \mathsf{tm}\end{array}}{\Gamma; \Psi \Vdash M = N : \mathsf{tm}}$$

$$\dfrac{\Gamma; \Psi \vdash M \searrow_{\mathsf{LF}} x : \mathsf{tm} \quad \Gamma; \Psi \vdash N \searrow_{\mathsf{LF}} x : \mathsf{tm}}{\Gamma; \Psi \Vdash M = N : \mathsf{tm}}$$

Fig. 16. Semantic Equality for LF Terms: $\boxed{\Gamma; \Psi \Vdash M = N : A}$

We start by defining semantic equality for LF terms of type tm (Fig. 16), as we restricted our LF signature and these are the terms of interest. To define semantic equality for LF terms M and N, we consider different cases depending on their whnf: 1) if they reduce to $\mathsf{app}\ M_1\ M_2$ and $\mathsf{app}\ N_1\ N_2$ respectively, then M_i must be semantically equal to N_i; 2) if they reduce to $\mathtt{lam}\ M'$ and $\mathtt{lam}\ N'$ respectively, then the bodies of M' and N' must be equal. To compare their bodies, we apply both M' and N' to an LF variable x and consider $M'\ x$ and $N'\ x$ in the extended LF context $\Psi, x{:}\mathsf{tm}$. This has the effect of opening up the body and replacing the bound LF variable with a fresh one. This highlights the difference between the intensional LF function space and the extensional nature of the computation-level functions. In the former, we can concentrate on LF variables and continue to analyze the LF function body; in the latter, we consider all possible inputs, not just variables; 3) if the LF terms M and N may reduce to the same LF variable in Ψ, then they are obviously also semantically equal; 4) last, if M and N reduce to $\lfloor t_i \rfloor_{\sigma_i}$ respectively. In this case t_i is neutral and we only need to semantically compare the LF substitutions σ_i and check whether the terms t_i are definitional equal. However, what type should we choose? – As the computation t_i is neutral, we can infer a unique type $\lceil \Phi \vdash \mathsf{tm} \rceil$ which we can use. This is defined as follows:

Type inference for Neutral Computations t : $\mathsf{typeof}(\Gamma \vdash t) = \tau$

$$\frac{\mathsf{typeof}(\Gamma \vdash t) = \tau \quad \tau \searrow (y{:}\tau_1) \Rightarrow \tau_2 \quad \Gamma \vdash s : \tau_1}{\mathsf{typeof}(\Gamma \vdash t\ s) = \{s/y\}\tau_2} \qquad \frac{x{:}\tau \in \Gamma}{\mathsf{typeof}(\Gamma \vdash x) = \tau}$$

$$\frac{\mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (y : \lceil \psi \vdash \mathsf{tm} \rceil) \Rightarrow \tau}{\mathsf{typeof}(\Gamma \vdash \mathsf{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t) = \{\Psi/\psi, t/y\}\tau}$$

Lemma 5.1. *If* $\Gamma \vdash t : \tau$ *and* $\mathsf{wne}\ t$ *then* $\mathsf{typeof}(\Gamma \vdash t) = \tau'$ *and* $\Gamma \vdash \tau \equiv \tau' : u$.

Proof. By induction on $\mathsf{wne}\ t$. □

Semantic equality for LF substitutions is also defined by considering different weak head normal forms (see Fig. 17). As we only work with well-typed LF objects, there is only one inhabitant for an empty context. Moreover, given a LF substitution with domain $\Phi, x{:}A$, we can weak head reduce the LF substitutions σ and σ' and continue to recursively compare them. Finally, for LF substitutions with domain ψ, a context variable, there are two cases we consider: either both LF substitution reduce to a weakening wk_ψ or they reduce to substitution closure.

$$\frac{\Gamma; \Psi \vdash \sigma \searrow_{\mathsf{LF}} \cdot : \cdot \quad \Gamma; \Psi \vdash \sigma' \searrow_{\mathsf{LF}} \cdot : \cdot}{\Gamma; \Psi \Vdash \sigma = \sigma' : \cdot} \qquad \frac{\Gamma; \psi, \overrightarrow{x{:}A} \vdash \sigma \searrow_{\mathsf{LF}} \mathsf{wk}_\psi : \psi \quad \Gamma; \psi, \overrightarrow{x{:}A} \vdash \sigma' \searrow_{\mathsf{LF}} \mathsf{wk}_\psi : \psi}{\Gamma; \psi, \overrightarrow{x{:}A} \Vdash \sigma = \sigma' : \psi}$$

$$\frac{\begin{array}{l}\Gamma; \Psi \vdash \sigma \searrow_{\mathsf{LF}} \sigma_1, M : \Phi, x{:}A \\ \Gamma; \Psi \vdash \sigma' \searrow_{\mathsf{LF}} \sigma_2, N : \Phi, x{:}A \quad \Gamma; \Psi \Vdash \sigma_1 = \sigma_2 : \Phi \qquad \Gamma; \Psi \Vdash M = N : [\sigma_1/\hat{\Phi}]A\end{array}}{\Gamma; \Psi \Vdash \sigma = \sigma' : \Phi, x{:}A}$$

Fig. 17. Semantic Equality for LF Substitutions: $\boxed{\Gamma; \Psi \Vdash \sigma = \sigma' : \Phi}$

To keep the definition compact, we again overload the semantic kinding and equality for types and terms. For example, we define the judgment $\Gamma \Vdash \check{\tau} : u$ which falls into two parts: $\Gamma \Vdash \tau : u$ and $\Gamma \Vdash \mathsf{tm\_ctx} : u$ where the latter is simply notation, as $\mathsf{tm\_ctx}$ is not a computation-level type. Similarly, we define $\Gamma \Vdash t = t' : \check{\tau}$ to stand for $\Gamma \vdash t = t' : \tau$, i.e. semantic equality for terms, and semantic equality for LF contexts where we write $\Gamma \Vdash t = t' : \mathsf{tm\_ctx}$, although t and t' stand for LF contexts.

Our semantic kinding for types (Fig. 19) is used as a measure to define the semantic typing for computations. In particular, we define $\Gamma \Vdash \check{\tau} = \check{\tau}' : u$ and $\Gamma \Vdash t = t' : \check{\tau}$ recursively on the semantic kinding of $\check{\tau}$. i.e. $\Gamma \Vdash \check{\tau} : u$. For better readability, we simply write for example $\Gamma \Vdash t = t' : \lceil T \rceil$ instead of $\Gamma \Vdash t = t' : \tau$ where $\tau \searrow \lceil T \rceil$, and $\Gamma \vdash T \equiv T$ in proofs. We note that to prove reflexivity for types, we would need to strengthen our semantic kinding definition with the additional premise: $\forall \Gamma' \leq_\rho \Gamma.\ \Gamma' \Vdash s = s' : \{\rho\}\check{\tau}_1 \Longrightarrow \Gamma' \Vdash \{\rho, s/y\}\tau_2 = \{\rho, s'/y\}\tau_2 : u_2$. This is possible, but since semantic reflexivity for types is not needed, we keep the semantic kinding definition more compact.

$$\frac{\Gamma; \Psi \Vdash M = N : A}{\Gamma \Vdash_{\mathsf{LF}} (\hat{\Psi} \vdash M) = (\hat{\Psi} \vdash N) : (\Psi \vdash A)} \quad \frac{\Gamma; \Psi \Vdash_{\#} M = N : A}{\Gamma \Vdash_{\mathsf{LF}} (\hat{\Psi} \vdash M) = (\hat{\Psi} \vdash N) : (\Psi \vdash_{\#} A)}$$

Fig. 18. Semantic Typing for Contextual LF Terms

$$\frac{\Gamma \vdash \tau \searrow \lceil T \rceil : u \quad \Gamma \vdash T \equiv T}{\Gamma \Vdash \tau : u} \qquad \frac{\Gamma \vdash \tau \searrow u' : u \quad u' < u}{\Gamma \Vdash \tau : u} \qquad \frac{\Gamma \vdash \tau \searrow x\,\vec{t} : u \qquad \mathsf{wne}\ (x\,\vec{t})}{\Gamma \Vdash \tau : u}$$

$$\frac{\begin{array}{c}\Gamma \vdash \tau \searrow (y : \check{\tau}_1) \Rightarrow \tau_2 : u \quad \forall \Gamma' \leq_\rho \Gamma.\, \Gamma' \Vdash \{\rho\}\check{\tau}_1 : u_1 \\ \forall \Gamma' \leq_\rho \Gamma.\, \Gamma' \Vdash s = s\ : \{\rho\}\check{\tau}_1 \Longrightarrow \Gamma' \Vdash \{\rho, s/y\}\tau_2 : u\end{array}}{\Gamma \Vdash \tau : u} \; (u_1,\ u_2,\ u) \in \Re \qquad \frac{\vdash \Gamma}{\Gamma \Vdash \mathsf{tm\_ctx} : u}$$

Fig. 19. Semantic Kinding for Types $\boxed{\Gamma \Vdash \check{\tau} : u}$ (inductive)

Semantic Equality for Types: $\boxed{\Gamma \Vdash \check{\tau} = \check{\tau}' : u}$ recursively defined by recursion on $\Gamma \Vdash \tau : u$

$\Gamma \Vdash \mathsf{tm\_ctx} = \mathsf{tm\_ctx} : u \;:\Leftrightarrow\; true$

$\Gamma \Vdash \lceil T \rceil = \tau' : u \;:\Leftrightarrow\; \Gamma \vdash \tau' \searrow \lceil T' \rceil : u$ and $\Gamma \vdash T \equiv T'$

$\Gamma \Vdash u' = \tau' : u \;:\Leftrightarrow\; \Gamma \vdash \tau' \searrow u' : u$

$\Gamma \Vdash (y : \check{\tau}_1) \Rightarrow \tau_2 = \tau' : u \;:\Leftrightarrow\; \Gamma \vdash \tau' \searrow (y' : \check{\tau}_1') \Rightarrow \tau_2' : u$ and $\forall \Gamma' \leq_\rho \Gamma.\, \Gamma' \Vdash \{\rho\}\check{\tau}_1 = \{\rho\}\check{\tau}_1' : u_1$ and
$\forall \Gamma' \leq_\rho \Gamma.\, \Gamma' \Vdash s = s' : \{\rho\}\check{\tau}_1 \Longrightarrow \Gamma' \Vdash \{\rho, s/y\}\tau_2 = \{\rho, s'/y'\}\tau_2' : u_2$
where $(u_1, u_2, u) \in \Re$

$\Gamma \Vdash x\,\vec{t} = \tau' : u \;:\Leftrightarrow\; \Gamma \vdash \tau' \searrow x\,\vec{s} : u$ and $\Gamma \vdash x\,\vec{t} \equiv x\,\vec{s} : u$

Semantic Equality for Terms (Computations): $\boxed{\Gamma \Vdash t = t' : \check{\tau}}$ by recursion on $\Gamma \Vdash \check{\tau} : u$

$\Gamma \Vdash \Psi = \Psi' : \mathsf{tm\_ctx} \;:\Leftrightarrow\; \Gamma \vdash \Psi \equiv \Psi' : \mathsf{tm\_ctx}$

$\Gamma \Vdash t = t' : \lceil \Psi \vdash A \rceil \;:\Leftrightarrow\; \Gamma \vdash t \searrow w : \lceil \Psi \vdash A \rceil$ and $\Gamma \vdash t' \searrow w' : \lceil \Psi \vdash A \rceil$ and
$\Gamma; \Psi \Vdash \lfloor w \rfloor_{\mathsf{id}} = \lfloor w' \rfloor_{\mathsf{id}} : A$

$\Gamma \Vdash t = t' : u'$ — already defined above by recursion on $\Gamma \Vdash t : u'$
we note that we have $\Gamma \Vdash \tau : u$ where $\Gamma \vdash \tau \searrow u' : u$ and $u' < u$;
hence using the sem. eq. for types is well-founded as we do so at a smaller universe u'
and our sem. eq. for types is recursively defined on $\Gamma \Vdash t : u'$

$\Gamma \Vdash t = t' : (y : \check{\tau}_1) \Rightarrow \tau_2 \;:\Leftrightarrow\; \Gamma \vdash t \searrow w : (y : \check{\tau}_1) \Rightarrow \tau_2$ and $\Gamma \vdash t' \searrow w' : (y : \check{\tau}_1) \Rightarrow \tau_2$ and
$\forall \Gamma' \leq_\rho \Gamma.\, \Gamma' \Vdash s = s' : \{\rho\}\check{\tau}_1 \Longrightarrow \Gamma' \Vdash \{\rho\}w\ s = \{\rho\}w'\ s' : \{\rho, s/y\}\tau_2$

$\Gamma \Vdash t = t' : x\,\vec{s} \;:\Leftrightarrow\; \Gamma \vdash t \searrow n : x\,\vec{s}$ and $\Gamma \vdash t' \searrow n' : x\,\vec{s}$ and wne n, n' and $\Gamma \vdash n \equiv n' : x\,\vec{s}$

Fig. 20. Semantic Equality

Lemma 5.2 (Weakening of Type Inference for Neutral Computations). *If* $\mathrm{typeof}(\Gamma \vdash t) = \tau$ *and* $\Gamma' \leq_\rho \Gamma$ *then* $\mathrm{typeof}(\Gamma' \vdash \{\rho\}t) = \tau'$ *s.t.* $\tau' = \{\rho\}\tau$.

Proof. By induction on $\mathrm{typeof}(\Gamma \vdash t) = \tau$ using Lemma 4.4 (3). □

6 SEMANTIC PROPERTIES

6.1 Semantic Properties of LF

Lemma 6.1 (Well-Formedness of Semantic LF Typing).

(1) If $\Gamma;\Psi \Vdash M = N : A$ *then* $\Gamma;\Psi \vdash M : A$ *and* $\Gamma;\Psi \vdash N : A$ *and* $\Gamma \vdash M \equiv N : A$.
(2) If $\Gamma;\Psi \Vdash \sigma_1 = \sigma_2 : \Phi$ *then* $\Gamma;\Psi \vdash \sigma_1 : \Phi$ *and* $\Gamma;\Psi \vdash \sigma_2 : \Phi$ *and* $\Gamma;\Psi \vdash \sigma_1 \equiv \sigma_2 : \Phi$.

Proof. By induction on the semantic definition. In each case, we refer the Def. 4.3. To illustrate, consider the case where $\Gamma;\Psi \vdash M \searrow_{\mathsf{LF}} \lambda x.M' : \Pi x{:}A.B$ and $\Gamma;\Psi \vdash N \searrow_{\mathsf{LF}} \lambda x.N' : \Pi x{:}A.B$, we also know that $\Gamma;\Psi \vdash M \equiv \lambda x.M' : \Pi x{:}A.B$ and $\Gamma;\Psi \vdash N \equiv \lambda x.N' : \Pi x{:}A.B$ by Def. 4.3.

Further, we have that $\Gamma;\Psi, x{:}A \Vdash M' = N' : B$. By IH, we get that $\Gamma;\Psi, x{:}A \vdash M' \equiv N' : B$ By dec. equivalence rules, we have $\Gamma;\Psi \vdash \lambda x.M' \equiv \lambda x.N' : \Pi x{:}A.B$. Therefore, by symmetry and transitivity of $\equiv$, we have $\Gamma;\Psi \vdash M \equiv N : \Pi x{:}A.B$. The typing invariants are left implicit.
We show the expanded proofs below concentrating on showing $\equiv$ and leaving the tracking of typing invariants implicit.

Case. $$\frac{\Gamma;\Psi \vdash \sigma \searrow_{\mathsf{LF}} \cdot : \cdot \quad \Gamma;\Psi \vdash \sigma' \searrow_{\mathsf{LF}} \cdot : \cdot}{\Gamma;\Psi \Vdash \sigma = \sigma' : \cdot}$$

$\Gamma;\Psi \vdash \sigma_1 \equiv \cdot : \cdot$ and $\Gamma;\Psi \vdash \sigma_2 \equiv \cdot : \cdot$ — by Def. 4.3
$\Gamma;\Psi \vdash \sigma_1 \equiv \sigma_2 : \cdot$ — by symmetry and transitivity of $\equiv$

Case. $$\frac{\Gamma;\psi, \overrightarrow{x{:}A} \vdash \sigma \searrow_{\mathsf{LF}} \mathsf{wk}_{\psi} : \psi \quad \Gamma;\psi, \overrightarrow{x{:}A} \vdash \sigma' \searrow_{\mathsf{LF}} \mathsf{wk}_{\psi} : \psi}{\Gamma;\psi, \overrightarrow{x{:}A} \Vdash \sigma = \sigma' : \psi}$$

$\Gamma;\psi, \overrightarrow{x{:}A} \vdash \sigma_1 \equiv \mathsf{wk}_{\psi} : \psi$ and $\Gamma;\psi, \overrightarrow{x{:}A} \vdash \sigma_2 \equiv \mathsf{wk}_{\psi} : \psi$ — by Def. 4.3
$\Gamma;\psi\overrightarrow{x{:}A} \vdash \sigma_1 \equiv \sigma_2 : \psi$ — by symmetry and transitivity of $\equiv$

Case. $$\frac{\begin{array}{l}\Gamma;\Psi \vdash \sigma_1 \searrow_{\mathsf{LF}} \sigma_1', M : \Phi, x{:}A \\ \Gamma;\Psi \vdash \sigma_2 \searrow_{\mathsf{LF}} \sigma_2', N : \Phi, x{:}A \quad \Gamma;\Psi \Vdash \sigma_1' = \sigma_2' : \Phi \qquad \Gamma;\Psi \Vdash M = N : [\sigma_1'/\hat{\Phi}]A\end{array}}{\Gamma;\Psi \Vdash \sigma_1 = \sigma_2 : \Phi, x{:}A}$$

$\Gamma;\Psi \vdash \sigma_1 \equiv \sigma_1', M : \Phi, x{:}A$ and $\Gamma;\Psi \vdash \sigma_2 \equiv \sigma_2', N : \Phi, x{:}A$ — by Def. 4.3
$\Gamma;\Psi \vdash \sigma_1' \equiv \sigma_2' : \Phi$ and $\Gamma;\Psi \vdash M \equiv N : [\sigma_1'/\hat{\Phi}]A$ — by induction hypothesis
$\Gamma;\Psi \vdash \sigma_1', M \equiv \sigma_2', N : \Phi, x{:}A$ — by dec. equivalence rules
$\Gamma;\Psi \vdash \sigma_1 \equiv \sigma_2 : \Phi, x{:}A$ — by symmetry and transitivity of $\equiv$

Case. $$\frac{\begin{array}{l}\Gamma;\Psi \vdash M \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{\sigma_1} : \mathsf{tm} \quad \mathsf{typeof}(\Gamma \vdash t_1) = \lceil \Phi_1 \vdash \mathsf{tm} \rceil \quad \Gamma \vdash \Phi_1 \equiv \Phi_2 : \mathsf{ctx} \\ \Gamma;\Psi \vdash N \searrow_{\mathsf{LF}} \lfloor t_2 \rfloor_{\sigma_2} : \mathsf{tm} \quad \mathsf{typeof}(\Gamma \vdash t_2) = \lceil \Phi_2 \vdash \mathsf{tm} \rceil \quad \Gamma \vdash t_1 \equiv t_2 : \lceil \Phi_1 \vdash \mathsf{tm} \rceil \quad \Gamma;\Psi \Vdash \sigma_1 = \sigma_2 : \Phi_1\end{array}}{\Gamma;\Psi \Vdash M = N : \mathsf{tm}}$$

$\Gamma;\Psi \vdash M \equiv \lfloor t_1 \rfloor_{\sigma_1} : \mathsf{tm}$ and $\Gamma;\Psi \vdash N \equiv \lfloor t_2 \rfloor_{\sigma_2} : \mathsf{tm}$ — by Def. 4.3
$\Gamma;\Psi \vdash \sigma_1 \equiv \sigma_2 : \Phi_1$ — by induction hypothesis
$\Gamma;\Psi \vdash \lfloor t_1 \rfloor_{\sigma_1} \equiv \lfloor t_2 \rfloor_{\sigma_2} : \mathsf{tm}$ — by dec. equivalence rules
$\Gamma;\Psi \vdash M \equiv N : \mathsf{tm}$ — by symmetry and transitivity of $\equiv$

Case. $$\frac{\begin{array}{l}\Gamma;\Psi \vdash M \searrow_{\mathsf{LF}} \mathtt{lam}\, M' : \mathsf{tm} \\ \Gamma;\Psi \vdash N \searrow_{\mathsf{LF}} \mathtt{lam}\, N' : \mathsf{tm} \quad \Gamma;\Psi \Vdash M' = N' : \Pi x{:}\mathsf{tm}.\mathsf{tm}\end{array}}{\Gamma;\Psi \Vdash M = N : \mathsf{tm}}$$

$\Gamma;\Psi \vdash M \equiv \mathtt{lam}\, M' : \mathsf{tm}$ and $\Gamma;\Psi \vdash N \equiv \mathtt{lam}\, N' : \mathsf{tm}$ — by Def. 4.3
$\Gamma;\Psi \vdash M' \equiv N' : \Pi x{:}\mathsf{tm}.\mathsf{tm}$ — by induction hypothesis
$\Gamma;\Psi \vdash \mathtt{lam}\, M' \equiv \mathtt{lam}\, N' : \mathsf{tm}$ — by dec. equivalence rules
$\Gamma;\Psi \vdash M \equiv N : \mathsf{tm}$ — by symmetry and transitivity of $\equiv$

$$\textit{Case.}\quad \frac{\begin{array}{l}\Gamma;\Psi \vdash M \searrow_{\mathsf{LF}} \mathsf{app}\ M_1\ M_2 : \mathsf{tm} \\ \Gamma;\Psi \vdash N \searrow_{\mathsf{LF}} \mathsf{app}\ N_1\ N_2 : \mathsf{tm} \quad \Gamma;\Psi \Vdash M_1 = N_1 : \mathsf{tm} \quad \Gamma;\Psi \Vdash M_2 = N_2 : \mathsf{tm}\end{array}}{\Gamma;\Psi \Vdash M = N : \mathsf{tm}}$$

$\Gamma;\Psi \vdash M \equiv \mathsf{app}\ M_1\ M_2 : \mathsf{tm}$ and $\Gamma;\Psi \vdash N \equiv \mathsf{app}\ N_1\ N_2 : \mathsf{tm}$ — by Def. 4.3
$\Gamma;\Psi \vdash M_1 \equiv N_1 : \mathsf{tm}$ and $\Gamma;\Psi \vdash M_2 \equiv N_2 : \mathsf{tm}$ — by induction hypothesis
$\Gamma;\Psi \vdash \mathsf{app}\ M_1\ M_2 \equiv \mathsf{app}\ N_1\ N_2$ — by dec. equivalence rules
$\Gamma;\Psi \vdash M \equiv N : \mathsf{tm}$ — by symmetry and transitivity of $\equiv$

$$\textit{Case.}\quad \frac{\Gamma;\Psi \vdash M \searrow_{\mathsf{LF}} x : \mathsf{tm} \quad \Gamma;\Psi \vdash N \searrow_{\mathsf{LF}} x : \mathsf{tm}}{\Gamma;\Psi \Vdash M = N : \mathsf{tm}}$$

$\Gamma;\Psi \vdash M \equiv x : \mathsf{tm}$ and $\Gamma;\Psi \vdash N \equiv x : \mathsf{tm}$ — by Def. 4.3
$\Gamma;\Psi \vdash M \equiv N : \mathsf{tm}$ — by symmetry and transitivity of $\equiv$

□

Lemma 6.2 (Semantic Weakening for LF).

(1) If $\Gamma;\Psi \Vdash M = N : A$ *and* $\Gamma' \leq_\rho \Gamma$ *then* $\Gamma';\{\rho\}\Psi \Vdash \{\rho\}M = \{\rho\}N : \{\rho\}A$.
(2) If $\Gamma;\Psi \Vdash \sigma = \sigma' : \Phi$ *and* $\Gamma' \leq_\rho \Gamma$ *then* $\Gamma';\{\rho\}\Psi \Vdash \{\rho\}\sigma = \{\rho\}\sigma' : \{\rho\}\Phi$.

Proof. By induction on the first derivation.

$$\textit{Case.}\quad \frac{\begin{array}{l}\Gamma;\Psi \vdash M \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{\sigma_1} : \mathsf{tm} \quad \mathsf{typeof}(\Gamma \vdash t_1) = \lceil \Phi_1 \vdash \mathsf{tm} \rceil \quad \Gamma \vdash \Phi_1 \equiv \Phi_2 : \mathsf{ctx} \\ \Gamma;\Psi \vdash N \searrow_{\mathsf{LF}} \lfloor t_2 \rfloor_{\sigma_2} : \mathsf{tm} \quad \mathsf{typeof}(\Gamma \vdash t_2) = \lceil \Phi_2 \vdash \mathsf{tm} \rceil \quad \Gamma;\Psi \Vdash \sigma_1 = \sigma_2 : \Phi_1 \quad \Gamma \vdash t_1 \equiv t_2 : \lceil \Phi_1 \vdash \mathsf{tm} \rceil\end{array}}{\Gamma \Vdash M = N : \mathsf{tm}}$$

$\Gamma';\{\rho\}\Psi \vdash \{\rho\}M \searrow_{\mathsf{LF}} \{\rho\}(\lfloor t_1 \rfloor_{\sigma_1}) : \mathsf{tm}$ and $\Gamma';\{\rho\}\Psi \vdash \{\rho\}N \searrow_{\mathsf{LF}} \{\rho\}(\lfloor t_2 \rfloor_{\sigma_1}) : \mathsf{tm}$ — by Lemma 4.4
$\Gamma';\{\rho\}\Psi \vdash \{\rho\}M \searrow_{\mathsf{LF}} \lfloor \{\rho\}t_1 \rfloor_{\{\rho\}\sigma_1} : \mathsf{tm}$ and $\Gamma';\{\rho\}\Psi \vdash \{\rho\}N \searrow_{\mathsf{LF}} \lfloor \{\rho\}t_2 \rfloor_{\{\rho\}\sigma_2} : \mathsf{tm}$ — by subst. def.
$\Gamma';\{\rho\}\Psi \Vdash \{\rho\}\sigma_1 = \{\rho\}\sigma_2 : \{\rho\}\Phi_1$ — by IH
$\Gamma' \vdash \{\rho\}\Phi_1 \equiv \{\rho\}\Phi_2 : \mathsf{ctx}$ — by Lemma 3.18
$\Gamma' \vdash \{\rho\}t_1 \equiv \{\rho\}t_2 : \{\rho\}\lceil \Phi_1 \vdash \mathsf{tm} \rceil$ — by Lemma 3.18
$\Gamma' \vdash \{\rho\}t = \{\rho\}t' : \lceil \{\rho\}\Phi_1 \vdash \mathsf{tm} \rceil$ — by substitution def.
$\mathsf{typeof}(\Gamma' \vdash \{\rho\}t_1) = \lceil \{\rho\}\Phi_1 \vdash \mathsf{tm} \rceil$ — by Lemma 5.2
$\mathsf{typeof}(\Gamma' \vdash \{\rho\}t_2) = \lceil \{\rho\}\Phi_2 \vdash \mathsf{tm} \rceil$ — by Lemma 5.2
$\Gamma';\{\rho\}\Psi \Vdash \{\rho\}M = \{\rho\}N : \{\rho\}\mathsf{tm}$ — by rule and substitution def.

□

Lemma 6.3 (Backwards Closure for LF terms).

(1) If $\Gamma;\Psi \Vdash Q = N : A$ *and* $\Gamma;\Psi \vdash M \searrow_{\mathsf{LF}} Q : A$ *then* $\Gamma \Vdash M = N : A$
(2) If $\Gamma;\Psi \Vdash N = Q : A$ *and* $\Gamma;\Psi \vdash M \searrow_{\mathsf{LF}} Q : A$ *then* $\Gamma \Vdash N = M : A$

Proof. By case analysis on $\Gamma;\Psi \Vdash Q = N : A$ and the fact that Q is in whnf .

$$\textit{Case.}\quad \frac{\begin{array}{l}\Gamma;\Psi \vdash Q \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{\sigma_1} : \mathsf{tm} \quad \mathsf{typeof}(\Gamma \vdash t_1) = \lceil \Phi_1 \vdash \mathsf{tm} \rceil \quad \Gamma \vdash \Phi_1 \equiv \Phi_2 : \mathsf{ctx} \\ \Gamma;\Psi \vdash N \searrow_{\mathsf{LF}} \lfloor t_2 \rfloor_{\sigma_2} : \mathsf{tm} \quad \mathsf{typeof}(\Gamma \vdash t_2) = \lceil \Phi_2 \vdash \mathsf{tm} \rceil \quad \Gamma \vdash t_1 \equiv t_2 : \lceil \Phi_1 \vdash \mathsf{tm} \rceil \quad \Gamma;\Psi \Vdash \sigma_1 = \sigma_2 : \Phi_1\end{array}}{\Gamma;\Psi \Vdash Q = N : \mathsf{tm}}$$

$\Gamma;\Psi \vdash M \searrow_{\mathsf{LF}} Q : A$ — by assumption
$\mathsf{whnf}\ Q$ — by invariant of $\searrow_{\mathsf{LF}}$
$Q = \lfloor t_1 \rfloor_{\sigma_1}$ — since whnf Q

$\Gamma; \Psi \Vdash M = N : \mathsf{tm}$ using $\mathcal{M} \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{\sigma_1}$ and sem. def.

$$\text{Case.}\ \frac{\begin{array}{l}\Gamma; \Psi \vdash Q \searrow_{\mathsf{LF}} \mathtt{lam}\, M' : \mathsf{tm} \\ \Gamma; \Psi \vdash N \searrow_{\mathsf{LF}} \mathtt{lam}\, N' : \mathsf{tm} \quad \Gamma; \Psi \Vdash M' = N' : \Pi x{:}\mathsf{tm}.\mathsf{tm}\end{array}}{\Gamma; \Psi \Vdash Q = N : \mathsf{tm}}$$

$\Gamma; \Psi \vdash M \searrow_{\mathsf{LF}} Q : A$ by assumption
$\mathsf{whnf}\, Q$ by invariant of $\searrow_{\mathsf{LF}}$
$Q = \mathtt{lam}\, M'$ by $\Gamma; \Psi \vdash Q \searrow_{\mathsf{LF}} \mathtt{lam}\, M' : \mathsf{tm}$ using $\mathsf{whnf}\, Q$
$\Gamma; \Psi \Vdash M = N : \mathsf{tm}$ using $\Gamma; \Psi \vdash M \searrow_{\mathsf{LF}} \mathtt{lam}\, M' : \mathsf{tm}$

□

Lemma 6.4 (Semantic LF Equality is preserved under LF Substitution).

(1) If $\Gamma; \Psi \Vdash \sigma = \sigma' : \Phi$ and $\Gamma; \Phi \Vdash M = N : A$ then $\Gamma; \Psi \Vdash [\sigma/\hat{\Phi}]M = [\sigma'/\hat{\Phi}]N : [\sigma/\hat{\Phi}]A$.
(2) If $\Gamma; \Psi \Vdash \sigma = \sigma' : \Phi$ and $\Gamma; \Phi \Vdash \sigma_1 = \sigma_2 : \Phi'$ then $\Gamma; \Psi \Vdash [\sigma/\hat{\Phi}]\sigma_1 = [\sigma'/\hat{\Phi}]\sigma_2 : \Phi'$.

Proof. Proof by mutual induction on $\Gamma; \Phi \Vdash M = N : A$ and $\Gamma; \Phi \Vdash \sigma = \sigma' : \Phi$ using the fact that weak head reduction is preserved under substitution (Lemma 4.5).

(1) $\boxed{\text{If } \Gamma; \Psi \Vdash \sigma = \sigma' : \Phi \text{ and } \Gamma; \Phi \Vdash M = N : A \text{ then } \Gamma; \Psi \Vdash [\sigma/\hat{\Phi}]M = [\sigma'/\hat{\Phi}]N : [\sigma/\hat{\Phi}]A.}$

$$\text{Case.}\ \frac{\begin{array}{l}\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{\sigma_1} : \mathsf{tm} \quad \mathsf{typeof}(\Gamma \vdash t_1) = \lceil \Phi_1 \vdash \mathsf{tm} \rceil \quad \Gamma \vdash \Phi_1 = \Phi_2 : \mathsf{ctx} \\ \Gamma; \Phi \vdash N \searrow_{\mathsf{LF}} \lfloor t_2 \rfloor_{\sigma_2} : \mathsf{tm} \quad \mathsf{typeof}(\Gamma \vdash t_2) = \lceil \Phi_2 \vdash \mathsf{tm} \rceil \quad \Gamma; \Phi \Vdash \sigma_1 = \sigma_2 : \Phi_1 \quad \Gamma \vdash t_1 \equiv t_2 : \lceil \Phi_1 \vdash \mathsf{tm} \rceil\end{array}}{\Gamma; \Phi \Vdash M = N : \mathsf{tm}}$$

$\Gamma; \Psi \vdash \sigma : \Phi$ and $\Gamma; \Psi \vdash \sigma' : \Phi$ by well-formedness of semantic equ. (Lemma 6.1)
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\lfloor t_1 \rfloor_{\sigma_1}) : \mathsf{tm}$ by Lemma 4.5
$\Gamma; \Psi \vdash [\sigma'/\hat{\Phi}]N \searrow_{\mathsf{LF}} [\sigma'/\hat{\Phi}](\lfloor t_2 \rfloor_{\sigma_2}) : \mathsf{tm}$ by Lemma 4.5
$\Gamma; \Phi \Vdash [\sigma/\hat{\Phi}]\sigma_1 = [\sigma'/\hat{\Phi}]\sigma_2 : \Phi_1$ by IH
$\Gamma; \Phi \Vdash [\sigma/\hat{\Phi}]M = [\sigma'/\hat{\Phi}]N : \mathsf{tm}$ by well-typed

$$\text{Case.}\ \frac{\Gamma; \Phi \vdash M \searrow_{\mathsf{LF}} x : \mathsf{tm} \quad \Gamma; \Phi \vdash N \searrow_{\mathsf{LF}} x : \mathsf{tm}}{\Gamma; \Phi \Vdash M = N : \mathsf{tm}}$$

$\Gamma; \Psi \Vdash \sigma(x) = \sigma'(x) : \mathsf{tm}$ by $\Gamma; \Phi \Vdash \sigma = \sigma' : \Psi$
$\Gamma; \Psi \vdash \sigma(x) \searrow_{\mathsf{LF}} M' : \mathsf{tm}$ and $\Gamma; \Phi \vdash \sigma'(x) \searrow_{\mathsf{LF}} N' : \mathsf{tm}$ by $\Gamma; \Psi \Vdash \sigma(x) = \sigma'(x) : \mathsf{tm}$
$\Gamma; \Psi \Vdash M' = N' : \mathsf{tm}$ since both $\mathsf{whnf}\, M'$ and $\mathsf{whnf}\, N'$
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]M \searrow_{\mathsf{LF}} M' : \mathsf{tm}$ by Lemma 4.5
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]N \searrow_{\mathsf{LF}} N' : \mathsf{tm}$ by Lemma 4.5
$\Gamma; \Phi \Vdash [\sigma/\hat{\Phi}]M = [\sigma'/\hat{\Phi}]N : \mathsf{tm}$ Backwards Closure (Lemma 6.3)

Other cases are similar.

(2) $\boxed{\text{If } \Gamma; \Psi \Vdash \sigma = \sigma' : \Phi \text{ and } \Gamma; \Phi \Vdash \sigma_1 = \sigma_2 : \Phi' \text{ then } \Gamma; \Psi \Vdash [\sigma/\hat{\Phi}]\sigma_1 = [\sigma'/\hat{\Phi}]\sigma_2 : \Phi'.}$

Proof by induction on $\Gamma; \Phi \Vdash \sigma = \sigma' : \Phi$ using the fact that weak head reduction is preserved under substitution (Lemma 4.5).

$$\text{Case.}\ \frac{\Gamma; \Phi \vdash \sigma_1 \searrow_{\mathsf{LF}} \cdot : \cdot \quad \Gamma; \Phi \vdash \sigma_2 \searrow_{\mathsf{LF}} \cdot : \cdot}{\Gamma; \Phi \Vdash \sigma_1 = \sigma_2 : \cdot}$$

$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]\sigma_1 \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}]\cdot : \cdot$ by Lemma 4.5
$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]\sigma_2 \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}]\cdot : \cdot$ by Lemma 4.5
$[\sigma/\hat{\Phi}]\cdot = \cdot$ by LF subst. def.
$\Gamma\Psi \Vdash [\sigma/\hat{\Phi}]\sigma_1 = [\sigma/\hat{\Phi}]\sigma_2 : \cdot$ by sem. eq. def.

Case. $\dfrac{\Gamma; \phi, \overrightarrow{x{:}A} \vdash \sigma_1 \searrow_{\mathsf{LF}} \mathsf{wk}_{\phi} : \phi \qquad \Gamma; \phi, \overrightarrow{x{:}A} \vdash \sigma_2 \searrow_{\mathsf{LF}} \mathsf{wk}_{\phi} : \phi}{\Gamma; \phi, \overrightarrow{x{:}A} \Vdash \sigma_1 = \sigma_2 : \phi}$

$\Gamma; \Psi \vdash [\sigma/\phi, \vec{x}]\sigma_1 \searrow_{\mathsf{LF}} [\sigma/\phi, \vec{x}]\mathsf{wk}_{\phi} : \phi$ by Lemma 4.5
$[\sigma/\phi, \vec{x}]\mathsf{wk}_{\phi} = \mathrm{trunc}_{\phi}(\sigma/\phi, \vec{x}) = \sigma_1'$ where $\Gamma; \Psi \vdash \sigma_1' : \phi$ by LF subst. def.
$\Gamma; \Psi \vdash [\sigma'/\phi, \vec{x}]\sigma_2 \searrow_{\mathsf{LF}} [\sigma'/\phi, \vec{x}]\mathsf{wk}_{\phi} : \phi$ by Lemma 4.5
$[\sigma'/\phi, \vec{x}]\mathsf{wk}_{\phi} = \mathrm{trunc}_{\phi}(\sigma'/\phi, \vec{x}) = \sigma_2'$ where $\Gamma; \Psi \vdash \sigma_2' : \phi$ by LF subst. def.
$\Gamma; \Psi \Vdash \sigma_1' = \sigma_2' : \phi$ since $\Gamma; \Psi \Vdash \sigma = \sigma' : \phi, \vec{x}$

Case. $\dfrac{\begin{array}{l}\Gamma; \Phi \vdash \sigma_1 \searrow_{\mathsf{LF}} \sigma_1', M : \Phi', x{:}A \\ \Gamma; \Phi \vdash \sigma_2 \searrow_{\mathsf{LF}} \sigma_2', N : \Phi', x{:}A \quad \Gamma; \Phi \Vdash \sigma_1' = \sigma_2' : \Phi' \qquad \Gamma; \Psi \Vdash M = N : [\sigma_1'/\hat{\Phi}']A\end{array}}{\Gamma; \Phi \Vdash \sigma_1 = \sigma_2 : \Phi', x{:}A}$

$\Gamma; \Psi \vdash [\sigma/\hat{\Phi}]\sigma_1 \searrow_{\mathsf{LF}} [\sigma/\hat{\Phi}](\sigma_1', M) : \Phi', x{:}A$ by Lemma 4.5
$\Gamma; \Psi \vdash [\sigma'/\hat{\Phi}]\sigma_2 \searrow_{\mathsf{LF}} [\sigma'/\hat{\Phi}](\sigma_2', N) : \Phi', x{:}A$ by Lemma 4.5
$[\sigma'/\hat{\Phi}](\sigma_2', N) = [\sigma'/\hat{\Phi}]\sigma_2',\ [\sigma'/\hat{\Phi}]N$ by LF subst. def.
$[\sigma/\hat{\Phi}](\sigma_1', M) = [\sigma/\hat{\Phi}]\sigma_1',\ [\sigma/\hat{\Phi}]M$ by LF subst. def.
$\Gamma; \Psi \Vdash [\sigma/\hat{\Phi}]\sigma_1' = [\sigma'/\hat{\Phi}]\sigma_2' : \Phi'$ by IH
$\Gamma; \Psi \Vdash [\sigma/\hat{\Phi}]M = [\sigma'/\hat{\Phi}]N : [\sigma/\hat{\Phi}]([\sigma_1'/\hat{\Phi}']A)$ by IH
$\Gamma; \Psi[\sigma/\hat{\Phi}]\sigma_1 = [\sigma'/\hat{\Phi}]\sigma_2 : \Phi', x{:}A$ by sem. equ. def.

□

Lemma 6.5 (Semantic Weakening Substitution Exist).
If $\Gamma; \Psi, \overrightarrow{x{:}A} \vdash wk_{\hat{\Psi}} : \Psi$ *then* $\Gamma; \Psi, \overrightarrow{x{:}A} \Vdash wk_{\hat{\Psi}} = wk_{\hat{\Psi}} : \Psi$.

Proof. By induction on the LF context Ψ. *Case.* $\Psi = \cdot$.

$\Gamma; \cdot, \overrightarrow{x{:}A} \vdash \mathsf{wk}_{\cdot} : \cdot$ by assumption
$\Gamma; \cdot, \overrightarrow{x{:}A} \vdash \mathsf{wk}_{\cdot} \searrow \cdot : \cdot$ by $\searrow$ rule and typing
$\Gamma; \cdot, \overrightarrow{x{:}A} \Vdash \mathsf{wk}_{\cdot} = \mathsf{wk}_{\cdot} : \cdot$ by semantic. def.

Case. $\Psi = \Psi', y{:}B$

$\Gamma; \Psi', y{:}B, \overrightarrow{x{:}A} \vdash \mathsf{wk}_{\hat{\Psi}', y} : \Psi', y{:}B$ by assumption
$\Gamma; \Psi', y{:}B, \overrightarrow{x{:}A} \vdash \mathsf{wk}_{\hat{\Psi}'} : \Psi'$ by typing
$\Gamma; \Psi', y{:}B, \overrightarrow{x{:}A} \Vdash \mathsf{wk}_{\hat{\Psi}'} = \mathsf{wk}_{\hat{\Psi}'} : \Psi'$ by IH
$\Gamma; \Psi', y{:}B, \overrightarrow{x{:}A} \Vdash y = y : B$ by semantic eq. for LF terms, the fact that whnf x, and $B = [\mathsf{wk}_{\hat{\Psi}'}/\hat{\Psi}']B$

$\Gamma; \Psi', y{:}B, \overrightarrow{x{:}A} \vdash \mathsf{wk}_{\hat{\Psi}', y} \searrow \mathsf{wk}_{\hat{\Psi}'}, y : \Psi', y{:}B$ by $\searrow$ and typing rules

$\Gamma; \Psi', y{:}B, \overrightarrow{x{:}A} \vdash \mathsf{wk}_{\hat{\Psi}', y} = \mathsf{wk}_{\hat{\Psi}', y} : \Psi', y{:}B$ by sem. eq. for LF substitutions

Case. $\Psi = \psi$

$\Gamma; \psi, \overrightarrow{x{:}A} \vdash \mathsf{wk}_{\psi} : \psi$ by assumption

$\Gamma; \psi, \overrightarrow{x{:}A} \vdash \mathsf{wk}_{\psi} : \psi$ by $\searrow$ and typing and the fact that whnf wk_{ψ}

$\Gamma; \psi, \overrightarrow{x{:}A} \vdash \mathsf{wk}_{\psi} = \mathsf{wk}_{\psi} : \psi$ by sem. eq. for LF subst.

□

Lemma 6.6 (Semantic LF Context Conversion).

(1) If $\Gamma; \Psi, x{:}A_1 \Vdash M = N : B$ *and* $\Gamma; \Psi \vdash A_1 \equiv A_2 : \mathsf{type}$ *then* $\Gamma; \Psi, x{:}A_2 \Vdash M = N : B$

(2) If $\Gamma; \Psi, x{:}A_1 \Vdash \sigma = \sigma' : \Phi$ *and* $\Gamma; \Psi \vdash A_1 \equiv A_2 : \mathsf{type}$ *then* $\Gamma; \Psi, x{:}A_2 \Vdash \sigma = \sigma' : \Phi$.

Proof. The idea is to use $\Gamma; \Psi \vdash A_1 \equiv A_2 : \mathsf{type}$ and build LF weakening substitutions $\Gamma; \Psi, x{:}A_2, y{:}A_1 \Vdash \mathsf{wk}_{\hat{\Psi}}, y = \mathsf{wk}_{\hat{\Psi}}, y : \Psi, x{:}A_1$ and $\Gamma; \Psi, x{:}A_2 \Vdash \mathsf{wk}_{\hat{\Psi}}, x, x = \mathsf{wk}_{\hat{\Psi}}, x, x : \Psi, x{:}A_2, y{:}A_1$. Using semantic LF subst. (Lemma 6.4), we can then move $\Gamma; \Psi, x{:}A_1 \Vdash M = N : B$ to the new LF context $\Psi, x{:}A_2$.

(1): $\boxed{\text{If } \Gamma; \Psi, x{:}A_1 \Vdash M = N : B \text{ and } \Gamma; \Psi \vdash A_1 \equiv A_2 : \mathsf{type} \text{ then } \Gamma; \Psi, x{:}A_2 \Vdash M = N : B}$

$\Gamma \vdash \Psi : \mathsf{ctx}$ by Well-Formedness of Sem. LF Equ. (Lemma 6.1) and Well-formedness of LF context (Lemma 3.1)

$\Gamma \vdash \Psi, x{:}A_2 : \mathsf{ctx}$ by context well-formedness rules

$\Gamma; \Psi \vdash A_2 \equiv A_1 : \mathsf{type}$ by symmetry

$\Gamma; \Psi, x{:}A_2 \vdash A_2 \equiv A_1 : \mathsf{type}$ by LF weakening

$\Gamma; \Psi, x{:}A_2 \vdash x : A_2$ by typing rule using $\Gamma \vdash \Psi, x{:}A_2 : \mathsf{ctx}$

$\Gamma; \Psi, x{:}A_2 \vdash x : A_1$ conversion using $\Gamma; \Psi, x{:}A_2 \vdash A_2 \equiv A_1 : \mathsf{type}$

$\Gamma; \Psi, x{:}A_2 \vdash x : [\mathsf{wk}_{\hat{\Psi}} / \hat{\Psi}]A_1$ as $A_1 = [\mathsf{wk}_{\hat{\Psi}} / \hat{\Psi}]A_1$

$\Gamma; \Psi, x{:}A_2 \vdash \mathsf{wk}_{\hat{\Psi}}, x, x : \Psi, x{:}A_2, y{:}A_1$ by typing rules for LF substitution

$\Gamma; \Psi, x{:}A_2, y{:}A_1 \vdash \mathsf{wk}_{\hat{\Psi}}, y : \Psi, x{:}A_1$ by typing rule for LF substitution

$\Gamma; \Psi, x{:}A_2, y{:}A_1 \vdash \mathsf{wk}_{\hat{\Psi}} : \Psi$ by typing

$\Gamma; \Psi, x{:}A_2, y{:}A_1 \Vdash \mathsf{wk}_{\hat{\Psi}} = \mathsf{wk}_{\hat{\Psi}} : \Psi$ by Lemma 6.5

$\Gamma; \Psi, x{:}A_2, y{:}A_1 \Vdash \mathsf{wk}_{\hat{\Psi}}, y = \mathsf{wk}_{\hat{\Psi}}, y : \Psi, x{:}A_1$ by sem. equ. for LF subst. using the fact that whnf y

$\Gamma; \Psi, x{:}A_2 \Vdash \mathsf{wk}_{\hat{\Psi}} = \mathsf{wk}_{\hat{\Psi}} : \Psi$ by Lemma 6.5

$\Gamma; \Psi, x{:}A_2 \Vdash \mathsf{wk}_{\hat{\Psi}}, x, x = \mathsf{wk}_{\hat{\Psi}}, x, x : \Psi, x{:}A_2, y{:}A_1$ by sem. equ. for LF subst. using the fact that whnf x

$\Gamma; \Psi, x{:}A_2 \Vdash [\mathsf{wk}_{\hat{\Psi}}, x, x / \hat{\Psi}, x, y]M' = [\mathsf{wk}_{\hat{\Psi}}, x, x / \hat{\Psi}, x, y]N' : [\mathsf{wk}_{\hat{\Psi}}, x, x / \hat{\Psi}, x, y]B$

where $M' = [\mathsf{wk}_{\hat{\Psi}}, y / \hat{\Psi}, x]M$ and $N' = [\mathsf{wk}_{\hat{\Psi}}, y / \hat{\Psi}, x]N$ by semantic LF subst. (Lemma 6.4 twice)

$[\mathsf{wk}_{\hat{\Psi}}, x, x / \hat{\Psi}, x, y](\mathsf{wk}_{\hat{\Psi}}, y) = \mathsf{wk}_{\hat{\Psi}}, x$ by subst. def.

$\Gamma; \Psi, x{:}A_2 \vdash [\mathsf{wk}_{\hat{\Psi}}, x / \hat{\Psi}, x]M = [\mathsf{wk}_{\hat{\Psi}}, x / \hat{\Psi}, x]N : [\mathsf{wk}_{\hat{\Psi}}, x / \hat{\Psi}, x]B$ by previous lines

$\Gamma; \Psi, x{:}A_2 \Vdash M = N : B$ using the fact that $[\mathsf{wk}_{\hat{\Psi}}, x / \hat{\Psi}, x]M = M$, etc.

We prove (2): $\boxed{\text{If } \Gamma; \Psi, x{:}A_1 \Vdash \sigma = \sigma' : \Phi \text{ and } \Gamma; \Psi \vdash A_1 \equiv A_2 : \mathsf{type} \text{ then } \Gamma; \Psi, x{:}A_2 \Vdash \sigma = \sigma' : \Phi.}$

$\Gamma; \Psi, x{:}A_2, y{:}A_1 \Vdash \mathsf{wk}_{\hat{\Psi}}, y = \mathsf{wk}_{\hat{\Psi}}, y : \Psi, x{:}A_1$ constructed as for case (1)

$\Gamma; \Psi, x{:}A_2 \Vdash \mathsf{wk}_{\hat{\Psi}}, x, x = \mathsf{wk}_{\hat{\Psi}}, x, x : \Psi, x{:}A_2, y{:}A_1$ constructed as for case (1)

$\Gamma; \Psi, x{:}A_2 \Vdash [\mathsf{wk}_{\hat{\Psi}}, x, x/\hat{\Psi}, x, y]\sigma_1 = [\mathsf{wk}_{\hat{\Psi}}, x, x/\hat{\Psi}, x, y]\sigma_2 : \Phi$

where $\sigma_1 = [\mathsf{wk}_{\hat{\Psi}}, y/\hat{\Psi}, x]\sigma$ and $\sigma_2 = [\mathsf{wk}_{\hat{\Psi}}, y/\hat{\Psi}, x]\sigma'$ by semantic LF subst. (Lemma 6.4 twice)

$[\mathsf{wk}_{\hat{\Psi}}, x, x/\hat{\Psi}, x, y](\mathsf{wk}_{\hat{\Psi}}, y) = \mathsf{wk}_{\hat{\Psi}}, x$ by subst. def.

$\Gamma; \Psi, x{:}A_2 \vdash [\mathsf{wk}_{\hat{\Psi}}, x/\hat{\Psi}, x]\sigma = [\mathsf{wk}_{\hat{\Psi}}, x/\hat{\Psi}, x]\sigma' : \Phi$ by previous lines

$\Gamma; \Psi, x{:}A_2 \Vdash \sigma = \sigma' : \Phi$ using the fact that $[\mathsf{wk}_{\hat{\Psi}}, x/\hat{\Psi}, x]\sigma = \sigma$, etc.

□

Our semantic definitions are reflexive, symmetric, and transitive. Further they are stable under type conversions. Establishing these properties is tricky and intricate. We first establish these properties for LF and subsequently for computations. All proofs can be found in the long version.

Lemma 6.7 (Symmetry, Transitivity, and Conversion of Semantic Equality for LF).
A. For LF Terms:

1. *(Reflexivity:)* $\Gamma; \Psi \Vdash M = M : A$.
2. *(Symmetry:) If* $\Gamma; \Psi \Vdash M = N : A$ *then* $\Gamma; \Psi \Vdash N = M : A$.
3. *(Transitivity:) If* $\Gamma; \Psi \Vdash M_1 = M_2 : A$ *and* $\Gamma; \Psi \Vdash M_2 = M_3 : A$ *then* $\Gamma; \Psi \Vdash M_1 = M_3 : A$.
4. *(Conversion:) If* $\Gamma; \Psi \vdash A \equiv A' : \mathsf{type}$ *and* $\Gamma; \Psi \Vdash M = N : A$ *then* $\Gamma; \Psi \Vdash M = N : A'$.

B. For LF Substitutions:

1. *(Reflexivity:)* $\Gamma; \Psi \Vdash \sigma = \sigma : \Phi$.
2. *(Symmetry:) If* $\Gamma; \Psi \Vdash \sigma = \sigma' : \Phi$ *then* $\Gamma; \Psi \Vdash \sigma' = \sigma : \Phi$.
3. *(Transitivity:) If* $\Gamma; \Psi \Vdash \sigma_1 = \sigma_2 : \Phi$ *and* $\Gamma; \Psi \Vdash \sigma_2 = \sigma_3 : \Phi$ *then* $\Gamma; \Psi \Vdash \sigma_1 = \sigma_3 : \Phi$.
4. *(Conversion:) If* $\Gamma \vdash \Phi \equiv \Phi' : \mathsf{ctx}$ *and* $\Gamma; \Psi \Vdash \sigma = \sigma' : \Phi$ *then* $\Gamma \Vdash \sigma = \sigma' : \Phi'$.

Proof. Reflexivity follows directly from symmetry and transitivity. For LF terms and substitutions, we prove symmetry and conversion by induction on the derivation $\Gamma; \Psi \Vdash M = N : A$ and $\Gamma; \Psi \Vdash \sigma = \sigma' : \Phi$ respectively. For transitivity, we use lexicographic induction.

We reason by induction on semantic equivalence relation where we consider any σ' smaller than σ if $\sigma \searrow_{\mathsf{LF}} \sigma'$; the proofs is mostly straightforward exploiting symmetry of decl. equivalence ($\equiv$), determinacy of weak head reductions, and crucially relies on well-formedness of semantic equality (Lemma 6.1) and functionality of LF typing (Lemma 3.6) for the case where $\sigma_i \searrow_{\mathsf{LF}} \sigma_i', M_i$.

Transitivity: If $\Gamma; \Psi \Vdash \sigma_1 = \sigma_2 : \Phi$ and $\Gamma; \Psi \Vdash \sigma_2 = \sigma_3 : \Phi$ then $\Gamma; \Psi \Vdash \sigma_1 = \sigma_3 : \Phi$.

By lexicographic induction on the first wo derivations;

Case. $\dfrac{\Gamma; \Psi \vdash \sigma_1 \searrow_{\mathsf{LF}} \cdot : \cdot \quad \Gamma; \Psi \vdash \sigma_2 \searrow_{\mathsf{LF}} \cdot : \cdot}{\Gamma; \Psi \Vdash \sigma_1 = \sigma_2 : \cdot}$

$\Gamma; \Psi \Vdash \sigma_2 = \sigma_3 : \cdot$ by assumption

$\Gamma; \Psi \vdash \sigma_2 \searrow_{\mathsf{LF}} \cdot : \cdot$ by inversion and determinacy (Lemma 4.2)

$\Gamma; \Psi \vdash \sigma_3 \searrow_{\mathsf{LF}} \cdot : \cdot$ by inversion

$\Gamma; \Psi \Vdash \sigma_1 = \sigma_3 : \cdot$ using $\Gamma; \Psi \vdash \sigma_1 \searrow_{\mathsf{LF}} \cdot : \cdot$ and $\Gamma; \Psi \vdash \sigma_3 \searrow_{\mathsf{LF}} \cdot : \cdot$

Case. $\dfrac{\Gamma; \Psi, \overrightarrow{x{:}A} \vdash \sigma_1 \searrow_{\mathsf{LF}} \mathsf{wk}_{\hat{\Psi}} : \Psi \quad \Gamma; \Psi, \overrightarrow{x{:}A} \vdash \sigma_2 \searrow_{\mathsf{LF}} \mathsf{wk}_{\hat{\Psi}} : \Psi}{\Gamma; \Psi \Vdash \sigma_1 = \sigma_2 : \Psi}$

$\Gamma; \Psi \vdash \sigma_2 \searrow_{\mathsf{LF}} \cdot : \cdot$ by assumption

$\Gamma; \Psi, \overrightarrow{x{:}A} \vdash \sigma_2 \searrow_{\mathsf{LF}} \mathsf{wk}_{\hat{\Psi}} : \Psi$ by inversion and determinacy (Lemma 4.2)

$\Gamma; \Psi, \overrightarrow{x{:}A} \vdash \sigma_3 \searrow_{\mathsf{LF}} \mathsf{wk}_{\hat{\Psi}} : \Psi$ by inversion

$\Gamma \Vdash \sigma_1 = \sigma_3 : \cdot$ using $\Gamma; \Psi, \overrightarrow{x{:}A} \vdash \sigma_1 \searrow_{\mathsf{LF}} \mathsf{wk}_{\hat{\Psi}} : \Psi$ and $\Gamma; \Psi, \overrightarrow{x{:}A} \vdash \sigma_3 \searrow_{\mathsf{LF}} \mathsf{wk}_{\hat{\Psi}} : \Psi$

$$\textit{Case.}\ \frac{\begin{array}{l}\Gamma;\Psi\vdash\sigma_1 \searrow_{\mathsf{LF}} \sigma_1', M : \Phi, x{:}A \\ \Gamma;\Psi\vdash\sigma_2 \searrow_{\mathsf{LF}} \sigma_2', N : \Phi, x{:}A \quad \Gamma;\Psi\Vdash\sigma_1' = \sigma_2' : \Phi \qquad \Gamma;\Psi\Vdash M = N : [\sigma_1'/\hat{\Phi}]A\end{array}}{\Gamma;\Psi\Vdash\sigma_1 = \sigma_2 : \Phi, x{:}A}$$

$\Gamma;\Psi\Vdash\sigma_2 = \sigma_3 : \Phi, x{:}A$	by assumption
$\Gamma;\Psi\vdash\sigma_2 \searrow_{\mathsf{LF}} \sigma_2', N : \Phi, x{:}A$	by inversion and determinacy (Lemma 4.2)
$\Gamma;\Psi\vdash\sigma_3 \searrow_{\mathsf{LF}} \sigma_3', Q : \Phi, x{:}A$	by inversion
$\Gamma;\Psi\Vdash N = Q : [\sigma_2/\hat{\Phi}]A$	inversion
$\Gamma;\Psi\Vdash\sigma_2' = \sigma_3' : \Phi, x{:}A$	by inversion
$\Gamma;\Psi\Vdash\sigma_1' = \sigma_3' : \Phi, x{:}A$	by IH
$\Gamma;\Psi\vdash\sigma_2', N : \Phi, x{:}A$	by def. of well-typed whnf
$\Gamma\vdash\Phi, x{:}A : \mathsf{ctx}$	by well-formedness of LF typing
$\Gamma;\Phi\vdash A : \mathsf{type}$	by well-formedness of LF contexts
$\Gamma;\Psi\vdash\sigma_1 \equiv \sigma_2 : \Phi$	by well-formedness of semantic equality (Lemma 6.1)
$\Gamma;\Psi\vdash[\sigma_1/\hat{\Phi}]A \equiv [\sigma_2/\hat{\Phi}]A : \mathsf{type}$	by functionality of LF typing (Lemma 3.6)
$\Gamma;\Psi\Vdash N = Q : [\sigma_1/\hat{\Phi}]A$	by IH (Conversion 4)
$\Gamma;\Psi\Vdash M \equiv Q[\sigma_1/\hat{\Phi}]A$	by IH
$\Gamma;\Psi\Vdash\sigma_1 = \sigma_3 :: \Phi, x{:}A$	by sem. def.

$\boxed{\text{Symmetry: If } \Gamma;\Psi\Vdash\sigma_1 = \sigma_2 : \Phi \text{ then } \Gamma;\Psi\Vdash\sigma_2 = \sigma_1 : \Phi}$

By induction on semantic equivalence relation where we consider any σ' smaller than σ if $\sigma \searrow_{\mathsf{LF}} \sigma'$; the proof is mostly straightforward exploiting symmetry of decl. equivalence ($\equiv$), but also relies again on well-formedness of semantic equality (Lemma 6.1) and functionality of LF typing (Lemma 3.6) for the case where $\sigma_i \searrow_{\mathsf{LF}} \sigma_i', M_i$. We show the interesting case.

$$\textit{Case.}\ \frac{\begin{array}{l}\Gamma;\Psi\vdash\sigma_1 \searrow_{\mathsf{LF}} \sigma_1', M : \Phi, x{:}A \\ \Gamma;\Psi\vdash\sigma_2 \searrow_{\mathsf{LF}} \sigma_2', N : \Phi, x{:}A \quad \Gamma;\Psi\Vdash\sigma_1' = \sigma_2' : \Phi \qquad \Gamma;\Psi\Vdash M = N : [\sigma_1'/\hat{\Phi}]A\end{array}}{\Gamma;\Psi\Vdash\sigma_1 = \sigma_2 : \Phi, x{:}A}$$

$\Gamma;\Psi\Vdash\sigma_2' = \sigma_1' : \Phi$	by IH
$\Gamma\vdash\Phi, x{:}A : \mathsf{ctx}$	by well-formedness of typing
$\Gamma;\Phi\vdash A : \mathsf{type}$	by well-formedness of LF contexts
$\Gamma;\Psi\vdash\sigma_1' \equiv \sigma_2' : \Phi$	by well-formedness of semantic equality (Lemma 6.1)
$\Gamma;\Psi\vdash[\sigma_1/\hat{\Phi}]A \equiv [\sigma_2/\hat{\Phi}]A : \mathsf{type}$	by functionality of LF typing (Lemma 3.6)
$\Gamma;\Psi\Vdash M = N : [\sigma_2/\hat{\Phi}]A$	by IH (Conversion 4)
$\Gamma;\Psi\Vdash N = M : [\sigma_2/\hat{\Phi}]A$	by IH
$\Gamma;\Psi\Vdash\sigma_2 = \sigma_1 : \Phi, x{:}A$	by def. of semantic equivalence

We now consider some cases for establishing symmetry and transitivity for semantic equality of LF terms.

$$\textit{Case.}\ \frac{\begin{array}{c}\Gamma;\Psi \vdash M_1 \searrow_{\mathsf{LF}} \lfloor t_1 \rfloor_{\sigma_1} : \mathsf{tm} \quad \mathsf{typeof}(\Gamma \vdash t_1) = \lceil \Phi_1 \vdash \mathsf{tm} \rceil \quad \Gamma \vdash \Phi_1 \equiv \Phi_2 : \mathsf{ctx} \\ \Gamma;\Psi \vdash M_2 \searrow_{\mathsf{LF}} \lfloor t_2 \rfloor_{\sigma_2} : \mathsf{tm} \quad \mathsf{typeof}(\Gamma \vdash t_2) = \lceil \Phi_2 \vdash \mathsf{tm} \rceil \quad \Gamma \vdash t_1 \equiv t_2 : \lceil \Phi_1 \vdash \mathsf{tm} \rceil \quad \Gamma \Vdash \sigma_1' = \sigma_2' : \Phi_1\end{array}}{\Gamma;\Psi \Vdash M_1 = M_2 : \mathsf{tm}}$$

Symmetry for LF Terms.

$\Gamma;\Psi \Vdash \sigma_2' = \sigma_1' : \Phi_1$	by IH
$\Gamma;\Psi \Vdash \sigma_2' = \sigma_1' : \Phi_2$	by IH (Conversion (4))
$\Gamma \vdash \lceil \Phi_1 \vdash \mathsf{tm} \rceil \equiv \lceil \Phi_2 \vdash \mathsf{tm} \rceil : u$	since $\Gamma \vdash \Phi_1 \equiv \Phi_2 : ctx$
$\Gamma \vdash t_1 \equiv t_2 : \lceil \Phi_2 \vdash \mathsf{tm} \rceil$	by conversion using $\Gamma \vdash \lceil \Phi_1 \vdash \mathsf{tm} \rceil \equiv \lceil \Phi_2 \vdash \mathsf{tm} \rceil : u$
$\Gamma \vdash t_2 \equiv t_1 : \lceil \Phi_2 \vdash \mathsf{tm} \rceil$	by transitivity of $\equiv$
$\Gamma;\Psi \Vdash N = M : A$	by sem. def.

Transitivity for LF Terms.

$\Gamma \Vdash M_2 = M_3 : \mathsf{tm}$	by assumption
$\Gamma \vdash M_2 \searrow_{\mathsf{LF}} \lfloor t_2 \rfloor_{\sigma_2} : \mathsf{tm}$	by inversion and determinacy (Lemma 4.2)
$\Gamma \vdash M_3 \searrow_{\mathsf{LF}} \lfloor t_3 \rfloor_{\sigma_3}$	by inversion
$\mathsf{typeof}(\Gamma \vdash t_2) = \lceil \Phi_2 \vdash \mathsf{tm} \rceil$	by inversion and uniqueness of typeof
$\mathsf{typeof}(\Gamma \vdash t_3) = \lceil \Phi_3 \vdash \mathsf{tm} \rceil$	by inversion
$\Gamma \vdash \Phi_2 \equiv \Phi_3 : \mathsf{ctx}$	by inversion
$\Gamma \vdash \Phi_1 \equiv \Phi_3 : \mathsf{ctx}$	by transitivity ($\equiv$)
$\Gamma \Vdash \sigma_2 = \sigma_3 : \Phi_2$	by inversion
$\Gamma \Vdash \sigma_2 = \sigma_3 : \Phi_1$	by IH (Conversion 4 using $\Gamma \Vdash \Phi_2 = \Phi_1 : \mathsf{ctx}$)
$\Gamma \Vdash \sigma_1 = \sigma_3 : \Phi_1$	by IH
$\Gamma \vdash t_2 \equiv t_3 : \lceil \Phi_2 \vdash \mathsf{tm} \rceil$	by inversion on $\Gamma;\Psi \Vdash M_2 = M_3 : \mathsf{tm}$
$\Gamma \vdash (\Phi_1 \vdash \mathsf{tm}) \equiv (\Phi_2 \vdash \mathsf{tm}) : u$	since $\Gamma \vdash \Phi_1 \equiv \Phi_2 : ctx$
$\Gamma \vdash t_2 \equiv t_3 : \lceil \Phi_1 \vdash \mathsf{tm} \rceil$	by type conversion using $\Gamma \vdash (\Phi_1 \vdash \mathsf{tm}) \equiv (\Phi_2 \vdash \mathsf{tm}) : u$)
$\Gamma \Vdash t_1 \equiv t_3 : \lceil \Phi_1 \vdash \mathsf{tm} \rceil$	by Transitivity of $\equiv$
$\Gamma \Vdash M_1 = M_3 : \mathsf{tm}$	

We concentrate here on proving the conversion properties:

$\boxed{\text{(Conversion:) If } \Gamma;\Psi \vdash A \equiv A' : \mathsf{type} \text{ and } \Gamma;\Psi \Vdash M = N : A \text{ then } \Gamma;\Psi \Vdash M = N : A'.}$

$$\textit{Case.}\ \frac{\Gamma;\Psi \vdash M \searrow_{\mathsf{LF}} \lambda x.M' : \Pi x{:}A.B \quad \Gamma;\Psi \vdash N \searrow_{\mathsf{LF}} \lambda x.N' : \Pi x{:}A.B \quad \Gamma;\Psi, x{:}A \Vdash M' = N' : B}{\Gamma;\Psi \Vdash M = N : \Pi x{:}A.B}$$

$\Gamma;\Psi \vdash \Pi x{:}A.B \equiv \Pi x{:}A'.B' : \mathsf{type}$	by assumption
$\Gamma;\Psi, x{:}A \vdash B \equiv B' : \mathsf{type}$ and $\Gamma;\Psi \vdash A \equiv A' : \mathsf{type}$	by injectivity of Π-types (Lemma 3.8)
$\Gamma;\Psi, x{:}A \Vdash M' = N' : B'$	by IH
$\Gamma;\Psi, x{:}A' \Vdash M' = N' : B'$	by Semantic LF context conversion (Lemma 6.6)
$\Gamma;\Psi \Vdash M = N : \Pi x{:}A'.B'$	by sem. def.

Other cases are trivial since they are at type tm. □

6.2 Semantic Properties of Computations

Lemma 6.8 (Well-Formedness of Semantic Typing). *If* $\Gamma \Vdash t = t' : \check{\tau}$ *then* $\Gamma \vdash t : \check{\tau}$ *and* $\Gamma \vdash t' : \check{\tau}$ *and* $\Gamma \vdash t \equiv t' : \check{\tau}$.

Proof. By induction on the induction on $\Gamma \Vdash \check{\tau} : u$. . In each case, we refer the Def. 4.3. ☐

Lemma 6.9 (Semantic Weakening for Computations).

(1) If $\Gamma \Vdash \check{\tau} : u$ *and* $\Gamma' \leq_\rho \Gamma$ *then* $\Gamma' \Vdash \{\rho\}\check{\tau} : u$.
(2) If $\Gamma \Vdash \check{\tau} = \check{\tau}' : u$ *and* $\Gamma' \leq_\rho \Gamma$ *then* $\Gamma' \Vdash \{\rho\}\check{\tau} = \{\rho\}\check{\tau}' : u$.
(3) If $\Gamma \Vdash t = t' : \check{\tau}$ *and* $\Gamma' \leq_\rho \Gamma$ *then* $\Gamma' \Vdash \{\rho\}t = \{\rho\}t : \{\rho\}\check{\tau}$.

Proof. By induction on $\Gamma \Vdash \check{\tau} : u$.

We note that the theorem is trivial for $\check{\tau} = \mathsf{tm\_ctx}$. Hence we concentrate on proving it where $\check{\tau} = \tau$ (i.e. it is a proper type).

For better and easier readability we simply write for example $\tau = (y : \check{\tau}_1) \Rightarrow \tau_2$ instead of
$\Gamma \Vdash \tau : u$ where

(1) $\Gamma \vdash \tau \searrow (x : \check{\tau}_1) \Rightarrow \tau_2$
(2) $(\forall \Gamma' \leq_\rho \Gamma . \Gamma' \Vdash \{\rho\}\check{\tau}_1 : u_1)$
(3) $\forall \Gamma' \leq_\rho \Gamma .\ \Gamma' \Vdash s = s' : \{\rho\}\check{\tau}_1 \Longrightarrow \Gamma' \Vdash \{\rho, s/x\}\tau_2 = \{\rho, s'/x\}\tau_2 : u_2$ and
(4) $(u_1, u_2, u_3) \in \mathcal{R}$.

Weakening of semantic typing $\Gamma \Vdash \tau : u$:
By case analysis on $\Gamma \Vdash \tau : u$.

Case. $\tau = (x : \check{\tau}_1) \Rightarrow \tau_2$

$\Gamma' \vdash \{\rho\}\tau \searrow \{\rho\}((y : \check{\tau}_1) \Rightarrow \tau_2) : u_3$ by Lemma 4.4 using $\Gamma \Vdash \tau : u$
$\Gamma' \vdash \{\rho\}\tau \searrow (x : \{\rho\}\check{\tau}_1) \Rightarrow \{\rho, x/x\}\tau_2 : u_3$ by substitution def.

Suppose that $\Gamma_1 \leq_{\rho_1} \Gamma'$
$\Gamma' \leq_\rho \Gamma$ by assumption
$\Gamma_1 \leq_{\{\rho_1\}\rho} \Gamma$ by composition of substitution
$\Gamma_1 \Vdash \{\{\rho_1\}\rho\}\check{\tau}_1 : u_1$ by $\Gamma \Vdash \tau : u$
$\Gamma_1 \Vdash \{\rho_1\}\{\rho\}\check{\tau}_1 : u_1$ by composition of substitution

Suppose that $\Gamma_1 \leq_{\rho_1} \Gamma'$ and $\Gamma_1 \Vdash s = s' : \{\rho_1\}(\{\rho\}\check{\tau}_1)$
$\Gamma_1 \leq_{\{\rho_1\}\rho} \Gamma$ by composition of substitution
$\Gamma_1 \Vdash s = s' : \{\{\rho_1\}\rho\}\check{\tau}_1$ by composition of substitution
$\Gamma_1 \Vdash \{\{\rho_1\}\rho, s/x\}\tau_2 = \{\{\rho_1\}\rho, s'/x\}\tau_2 : u_2$ by $\Gamma \Vdash \tau : u$
$\Gamma_1 \Vdash \{\rho_1, s/x\}(\{\rho, x/x\}\tau_2) = \{\rho_1, s'/x\}(\{\rho, x/x\}\tau_2) : u_2$ by composition of substitution

$\Gamma' \Vdash \{\rho\}\tau : u_3$ by abstraction, since Γ_1, ρ_1 where arbitrary

Case. $\tau = \lceil T \rceil$

$\Gamma' \vdash \{\rho\}\tau \searrow \{\rho\}\lceil T \rceil : u$ by Lemma 4.4 using $\Gamma \Vdash \tau : u$
$\Gamma' \vdash \{\rho\}\tau \searrow \lceil \{\rho\}T \rceil : u$ by substitution def.
$\Gamma \Vdash_{\mathsf{LF}} T = T : u$ by assumption
$\Gamma \vdash T \equiv T : u$ by inversion
$\Gamma' \vdash \{\rho\}T \equiv \{\rho\}T : u$ by substitution lemma
$\Gamma' \Vdash \{\rho\}\tau : u$ by def.

Case. $\Gamma \Vdash t = t' : \lceil \Psi \vdash A \rceil$

$\Gamma \vdash t \searrow w : \lceil \Psi \vdash A \rceil$ by assumption
$\Gamma' \vdash \{\rho\}t \searrow \{\rho\}w : \{\rho\}\lceil \Psi \vdash A \rceil$ by 4.4
$\Gamma \vdash t' \searrow w' : \lceil \Psi \vdash A \rceil$ by assumption
$\Gamma' \vdash \{\rho\}t' \searrow \{\rho\}w' : \{\rho\}\lceil \Psi \vdash A \rceil$ by 4.4
$\Gamma; \Psi \Vdash \lfloor w \rfloor_{\mathsf{id}} = \lfloor w' \rfloor_{\mathsf{id}} : A$ by assumption
$\Gamma'; \{\rho\}\Psi \Vdash \{\rho\}\lfloor w \rfloor_{\mathsf{id}} = \{\rho\}\lfloor w' \rfloor_{\mathsf{id}} : \{\rho\}A$ by IH
$\Gamma' \Vdash \{\rho\}t = \{\rho\}t' : \{\rho\}\lceil \Psi \vdash A \rceil$ by sem. def. using subst. properties

Case. $\Gamma \Vdash t = t' : (y : \check{\tau}_1) \Rightarrow \tau_2$

$\Gamma \vdash t \searrow w : (y : \check{\tau}_1) \Rightarrow \tau_2$ by assumption
$\Gamma' \vdash \{\rho\}t \searrow \{\rho\}w : \{\rho\}((y : \check{\tau}_1) \Rightarrow \tau_2)$ by Lemma 4.4
$\Gamma \vdash t' \searrow w' : (y : \check{\tau}_1) \Rightarrow \tau_2$ by assumption
$\Gamma' \vdash \{\rho\}t' \searrow \{\rho\}w' : \{\rho\}((y : \check{\tau}_1) \Rightarrow \tau_2)$ by Lemma 4.4
Suppose that $\Gamma_1 \leq_{\rho_1} \Gamma'$ and $\Gamma_1 \vdash t_1 : \{\rho_1\}(\{\rho\}\check{\tau}_1)$ and $\Gamma_1 \vdash t_2 : \{\rho_1\}(\{\rho\}\check{\tau}_1)$
$\Gamma_1 \leq_{\{\rho_1\}\rho} \Gamma'$ by definition
$\Gamma_1 \vdash t_1 : \{\{\rho_1\}\{\rho\}\check{\tau}_1$ by composition of substitution
$\Gamma_1 \vdash t_2 : \{\{\rho_1\}\rho\}\check{\tau}_1$ by composition of substitution
$\Gamma_1 \Vdash t_1 = t_2 : \{\{\rho_1\}\rho\}\check{\tau}_1 \Longrightarrow \Gamma_1 \Vdash \{\{\rho_1\}\rho\}w\ t_1 = \{\{\rho_1\}\rho\}w'\ t_2 : \{\{\rho_1\}\rho, t_1/x\}\tau_2$ by assumption
$\Gamma_1 \Vdash t_1 = t_2 : \{\rho_1\}(\{\rho\}\check{\tau}_1) \Longrightarrow \Gamma_1 \Vdash (\{\rho_1\}(\{\rho\}w))\ t_1 = (\{\rho_1\}\{\rho\}w')\ t_2 : \{\rho_1, t_1/x\}(\{\rho\}\tau_2)$ by composition of substitution
$\Gamma' \Vdash \{\rho\}t = \{\rho\}t' : \{\rho\}((y : \check{\tau}_1) \Rightarrow \tau_2)$ by def.

□

Our semantic equality definition is symmetric and transitive. Our semantic equality is also reflexive – however, note we prove a weaker reflexivity statement which says that if t_1 is semantically equivalent to another term t_2 then it is also equivalent to itself. This suffices for our proofs. We also note that our semantic equality takes into account extensionality for terms at function types and contextual types; this is in fact baked into our semantic equality definition.

Lemma 6.10 (Symmetry, Transitivity, and Conversion of Semantic Equality).
Let $\Gamma \Vdash \check{\tau} : u$ *and* $\Gamma \Vdash \check{\tau}' : u$ *and* $\Gamma \Vdash \check{\tau} = \check{\tau}' : u$ *and* $\Gamma \Vdash t_1 = t_2 : \check{\tau}$. *Then:*

(1) *(Reflexivity for Terms:)* $\Gamma \Vdash t_1 = t_1 : \check{\tau}$.
(2) *(Symmetry for Terms:)* $\Gamma \Vdash t_2 = t_1 : \check{\tau}$.
(3) *(Transitivity for Terms:) If* $\Gamma \Vdash t_2 = t_3 : \check{\tau}$ *then* $\Gamma \Vdash t_1 = t_3 : \check{\tau}$.
(4) *(Symmetry for Types:)* $\Gamma \Vdash \check{\tau}' = \check{\tau} : u$
(5) *(Transitivity for Types:) If* $\Gamma \Vdash \check{\tau}' = \check{\tau}'' : u$ *and* $\Gamma \Vdash \check{\tau}'' : u$ *then* $\Gamma \Vdash \check{\tau} = \check{\tau}'' : u$.
(6) *(Conversion:)* $\Gamma \Vdash t_1 = t_2 : \check{\tau}'$.

Proof. Reflexivity follows directly from symmetry and transitivity. We prove symmetry and transitivity for terms using a lexicographic induction on u and $\Gamma \Vdash \tau : u$; we appeal to the induction hypothesis and use the corresponding properties on types if the universe is smaller; if the universe stays the same, then we may appeal to the property for terms if $\Gamma \Vdash \tau : u$ is smaller;

To prove conversion and symmetry for types, we may also appeal to the induction hypothesis if $\Gamma \Vdash \tau' : u$ is smaller.

Case. $\boxed{\boxed{\check{\tau} = \mathsf{tm\_ctx}}}$

Symmetry for Terms (Prop. 2): To Show: $\boxed{\Gamma \Vdash t_2 = t_1 : \mathsf{tm\_ctx}}$
t_2 and t_1 stand for LF context Ψ_2 and Ψ_1 respectively

$\Gamma \vdash \Psi_1 \equiv \Psi_2 : \mathsf{tm\_ctx}$ by def. $\Gamma \Vdash t_1 = t_2 : \tau$
$\Gamma \vdash \Psi_2 \equiv \Psi_1 : \mathsf{tm\_ctx}$ by symmetry of $\equiv$
$\Gamma \Vdash t_2 = t_2 : \mathsf{tm\_ctx}$ by sem. equ. def.

Transitivity for Terms (Prop. 3): To Show: $\boxed{\text{If } \Gamma \Vdash t_2 = t_3 : \mathsf{tm\_ctx} \text{ then } \Gamma \Vdash t_1 = t_3 : \mathsf{tm\_ctx}.}$
t_1, t_2, and t_3 stand for LF context Ψ_1, Ψ_2, and Ψ_3 respectively
$\Gamma \vdash \Psi_1 \equiv \Psi_2 : \mathsf{tm\_ctx}$ by $\Gamma \Vdash t_1 = t_2 : \mathsf{tm\_ctx}$
$\Gamma \vdash \Psi_2 \equiv \Psi_3 : \mathsf{tm\_ctx}$ by $\Gamma \Vdash t_2 = t_3 : \mathsf{tm\_ctx}$
$\Gamma \vdash \Psi_1 \equiv \Psi_3 : \mathsf{tm\_ctx}$ by transitivity of $\equiv$
$\Gamma \Vdash t_1 = t_3 : \mathsf{tm\_ctx}$ by semantic equ. def.

Other cases are trivial.

Case. $\boxed{\boxed{\tau = \lceil T \rceil, \text{ i.e. } \Gamma \vdash \tau \searrow \lceil T \rceil : u \text{ and } \Gamma \Vdash_{\mathsf{LF}} T = T \text{ where } T = \Psi \vdash A}}$

Symmetry for Terms (Prop. 2): To Show: $\boxed{\Gamma \Vdash t_2 = t_1 : \lceil T \rceil.}$
$\Gamma \vdash t_1 \searrow w_1 : \tau$ by definition of $\Gamma \Vdash t_1 = t_2 : \tau$
$\Gamma \vdash t_2 \searrow w_2 : \tau$ by definition of $\Gamma \Vdash t_1 = t_2 : \tau$
Sub-Case: $\Gamma \vdash t_1 \searrow \lceil C \rceil : \lceil T \rceil$ and $\Gamma \vdash t_2 \searrow \lceil C' \rceil : \lceil T \rceil$ and $\Gamma \Vdash_{\mathsf{LF}} C = C' : T$
Consider $C = (\hat{\Psi} \vdash M)$ and $C' = (\hat{\Psi} \vdash N)$ and $T = \Psi \vdash A$ (proof is the same for case $C = (\hat{\Psi} \vdash \sigma)$)
$\Gamma; \Psi \Vdash M = N : A$ by $\Gamma \Vdash_{\mathsf{LF}} C = C' : T$
$\Gamma; \Psi \Vdash N = M : A$ by Lemma 6.7 (2)
$\Gamma \Vdash_{\mathsf{LF}} C' = C : T$ by sem. def.
$\Gamma \Vdash t_2 = t_1 : \lceil T \rceil$ by semantic equ. def.

Sub-Case: wne w_1, w_2 and $\Gamma \vdash w_1 \equiv w_2 : \lceil T \rceil$
$\Gamma \vdash w_2 \equiv w_1 : \lceil T \rceil$ by symmetry of $\equiv$
$\Gamma \Vdash t_2 = t_1 : \lceil T \rceil$ by semantic equ. def.

Transitivity for Terms (Prop.3): To Show $\boxed{\text{If } \Gamma \Vdash t_2 = t_3 : \lceil T \rceil \text{ then } \Gamma \Vdash t_1 = t_3 : \lceil T \rceil.}$
$\Gamma \vdash t_1 \searrow w_1 : \tau$ by definition of $\Gamma \Vdash t_1 = t_2 : \tau$
$\Gamma \vdash t_2 \searrow w_2 : \tau$ by definition of $\Gamma \Vdash t_1 = t_2 : \tau$
$\Gamma \vdash t_2 \searrow w_2' : \tau$ by definition of $\Gamma \Vdash t_2 = t_3 : \tau$
$\Gamma \vdash t_3 \searrow w_3 : \tau$ by definition of $\Gamma \Vdash t_2 = t_3 : \tau$
$w_2 = w_2'$ by Lemma 4.2 (3)
$\Gamma; \Psi \Vdash \lfloor w_1 \rfloor_{\mathsf{id}} = \lfloor w_2 \rfloor_{\mathsf{id}} : A$ by def. $\Gamma \Vdash t_1 = t_2 : \tau$
$\Gamma; \Psi \Vdash \lfloor w_2 \rfloor_{\mathsf{id}} = \lfloor w_3 \rfloor_{\mathsf{id}} : A$ by def. $\Gamma \Vdash t_2 = t_3 : \tau$
$\Gamma; \Psi \Vdash \lfloor w_1 \rfloor_{\mathsf{id}} = \lfloor w_3 \rfloor_{\mathsf{id}} : A$ by Lemma 6.7 (3)
$\Gamma \Vdash t_1 = t_3 : \lceil \Psi \vdash A \rceil$ by sem. equ. def.

Symmetry for Types (Prop.4): To Show: $\boxed{\Gamma \Vdash \tau' = \lceil T \rceil : u \text{ where } T = \Psi \vdash A}$
$\Gamma \vdash \tau' \searrow \lceil T' \rceil : u$ and $\Gamma \vdash T \equiv T'$ by $\Gamma \Vdash \tau = \tau' : u$
$\Gamma \vdash T' \equiv T$ by symmetry for LF equ.
$\Gamma \Vdash \tau' = \tau : u$ by semantic equ. def.

Transitivity for Types (Prop.5): To Show: $\boxed{\text{If } \Gamma \Vdash \tau' = \tau'' : u \text{ and } \Gamma \Vdash \tau'' : u \text{ then } \Gamma \Vdash \lceil T \rceil = \tau'' : u.}$

$\Gamma \vdash \tau' \searrow \lceil T' \rceil : u$ and $\Gamma \vdash T \equiv T'$ — by $\Gamma \Vdash \tau = \tau' : u$

$\Gamma \vdash \tau'' \searrow \lceil T'' \rceil : u$ and $\Gamma \vdash T' \equiv T''$ — by $\Gamma \Vdash \tau' = \tau'' : u$

$\Gamma \vdash T \equiv T''$ — by transitivity for LF equ.

$\Gamma \Vdash \tau = \tau'' : u$ — by semantic equ. def.

Conversion for Terms (Prop.6): To Show: $\boxed{\Gamma \Vdash t_1 = t_2 : \tau'.}$

$\Gamma \vdash t_1 \searrow w_1 : \lceil T \rceil$ — by definition of $\Gamma \Vdash t_1 = t_2 : \tau$

$\Gamma \vdash t_2 \searrow w_2 : \lceil T \rceil$ — by definition of $\Gamma \Vdash t_1 = t_2 : \tau$

$\Gamma \vdash \tau' \searrow \lceil T' \rceil : u$ and $\Gamma \vdash T \equiv T'$ — by def. of $\Gamma \Vdash \tau = \tau' : u$

$\Gamma \vdash \lceil T \rceil \equiv \lceil T' \rceil : u$ — by decl. equ. def.

$\Gamma \vdash \tau \equiv \lceil T \rceil : u$ — by $\Gamma \vdash \tau \searrow \lceil T \rceil$ (since $\searrow$ rules are a subset of $\equiv$)

$\Gamma \vdash \tau' \equiv \lceil T' \rceil : u$ — by $\Gamma \vdash \tau' \searrow \lceil T' \rceil$ (since $\searrow$ rules are a subset of $\equiv$)

$\Gamma \vdash \tau \equiv \tau' : u$ — by transitivity and symmetry of decl. equality ($\equiv$)

$\Gamma \vdash t_i : \tau'$ and $\Gamma \vdash w_i : \tau'$ for $i = 1, 2$ — by typing rules using $\Gamma \vdash t_i : \lceil T \rceil$

$\Gamma \vdash t_1 \searrow w_1 : \tau'$ and $\Gamma \vdash t_2 \searrow w_2 : \tau'$ — by Def. 4.3

$\Gamma \Vdash t_1 = t_2 : \tau'$ — by semantic equ. def.

Case. $\boxed{\boxed{\tau = (y : \check{\tau}_1) \Rightarrow \tau_2 \text{ i.e. } \Gamma \vdash \tau \searrow (y : \check{\tau}_1) \Rightarrow \tau_2 : u}}$

Symmetry for Terms (Prop. 2): To Show: $\boxed{\Gamma \Vdash t_2 = t_1 : (y : \check{\tau}_1) \Rightarrow \tau_2}$

$\Gamma \vdash t_1 \searrow w_1 : \tau$ — by definition of $\Gamma \Vdash t_1 = t_2 : \tau$

$\Gamma \vdash t_2 \searrow w_2 : \tau$ — by definition of $\Gamma \Vdash t_1 = t_2 : \tau$

Assume $\Gamma' \leq_\rho \Gamma$ and $\Gamma' \Vdash s_2 = s_1 : \{\rho\}\check{\tau}_1$

$\Gamma' \Vdash \{\rho\}\check{\tau}_1 : u_1$ — by $\Gamma \Vdash \tau : u$

$\Gamma' \Vdash s_1 = s_2 : \{\rho\}\check{\tau}_1$ — by induction hypothesis (Prop. 2), symmetry

$\Gamma' \Vdash \{\rho\}w_1\ s_1 = \{\rho\}w_2\ s_2 : \{\rho, s_1/y\}\tau_2$ — by definition of $\Gamma \Vdash t_1 = t_2 : \tau$

$\Gamma' \Vdash \{\rho, s_1/y\}\tau_2 : u_2$ — by assumption $\Gamma \Vdash \tau : u$

$\Gamma' \Vdash \{\rho\}w_2\ s_2 = \{\rho\}w_1\ s_1 : \{\rho, s_1/y\}\tau_2$ — by induction hypothesis (Prop. 2), symmetry

$\Gamma \vdash \tau' \searrow (y : \check{\tau}_1') \Rightarrow \tau_2' : u$ — by definition of $\Gamma \Vdash \tau = \tau' : u$

$\Gamma' \Vdash \{\rho, s_1/y\}\tau_2 = \{\rho, s_2/y\}\tau_2' : u_2$ — by definition of $\Gamma \Vdash \tau = \tau' : u$

$\Gamma' \Vdash s_2 = s_2 : \{\rho\}\check{\tau}_1$ — by induction hypothesis (Prop. 1), reflexivity

$\Gamma' \Vdash \{\rho, s_2/y\}\tau_2 = \{\rho, s_2/y\}\tau_2' : u_2$ — by definition of $\Gamma \Vdash \tau = \tau' : u$

$\Gamma' \Vdash \{\rho, s_2/y\}\tau_2' = \{\rho, s_2/y\}\tau_2 : u_2$ — by induction hypothesis (Prop. 4), symmetry

$\Gamma' \Vdash \{\rho, s_1/y\}\tau_2 = \{\rho, s_2/y\}\tau_2 : u_2$ — by induction hypothesis (Prop. 5), transitivity

$\Gamma' \Vdash \{\rho\}w_2\ s_2 = \{\rho\}w_1\ s_1 : \{\rho, s_2/y\}\tau_2$ — by induction hypothesis (Prop. 6), conversion

$\Gamma \Vdash t_2 = t_1 : \tau$ — since Γ', ρ, s_2, s_1 were arbitrary

Transitivity for Terms (Prop. 3): $\boxed{\text{If } \Gamma \Vdash t_2 = t_3 : (y : \check{\tau}_1) \Rightarrow \tau_2 \text{ then } \Gamma \Vdash t_1 = t_3 : (y : \check{\tau}_1) \Rightarrow \tau_2.}$

$\Gamma \vdash t_1 \searrow w_1 : \tau$ — by definition of $\Gamma \Vdash t_1 = t_2 : \tau$

$\Gamma \vdash t_2 \searrow w_2 : \tau$ — by definition of $\Gamma \Vdash t_1 = t_2 : \tau$

$\Gamma \vdash t_2 \searrow w_2' : \tau$ — by definition of $\Gamma \Vdash t_2 = t_3 : \tau$

$\Gamma \vdash t_3 \searrow w_3 : \tau$ — by definition of $\Gamma \Vdash t_2 = t_3 : \tau$

$w_2 = w_2'$ — by determinacy of weak head evaluation (Lemma 4.2)

Assume $\Gamma' \leq_\rho \Gamma$ and $\Gamma' \Vdash s_1 = s_3 : \{\rho\}\check{\tau}_1$

$\Gamma' \Vdash \{\rho\}\check{\tau}_1 : u_1$	by $\Gamma \Vdash \tau : u$
$\Gamma' \Vdash s_1 = s_1 : \{\rho\}\check{\tau}_1$	by induction hypothesis (Prop. 1), reflexivity
$\Gamma' \Vdash \{\rho, s_1/y\}\tau_2 : u_2$	by assumption $\Gamma \Vdash \tau : u$
$\Gamma' \Vdash \{\rho\}w_1\ s_1 = \{\rho\}w_2\ s_1 : \{\rho, s_1/y\}\tau_2$	by assumption $\Gamma \Vdash t_1 = t_2 : \tau$
$\Gamma' \Vdash \{\rho\}w_2\ s_1 = \{\rho\}w_3\ s_3 : \{\rho, s_1/y\}\tau_2$	by assumption $\Gamma \Vdash t_2 = t_3 : \tau$
$\Gamma' \Vdash \{\rho\}w_1\ s_1 = \{\rho\}w_3\ s_3 : \{\rho, s_1/y\}\tau_2$	by induction hypothesis (Prop. 3), transitivity
$\Gamma \Vdash t_1 = t_3 : \tau$	since Γ', ρ, s_1, s_3 were arbitrary

Symmetry for Types (Prop. 4): $\boxed{\Gamma \Vdash \tau' = \tau : u}$

$\Gamma \Vdash (y : \check{\tau}_1) \Rightarrow \tau_2 = \tau' : u$	by assumption
$\Gamma \vdash \tau' \searrow (y : \check{\tau}_1') \Rightarrow \tau_2'$	by definition of $\Gamma \Vdash \tau = \tau' : u$
Assume $\Gamma' \leq_\rho \Gamma$ and $\Gamma' \Vdash s' = s : \{\rho\}\check{\tau}_1'$.	
$\Gamma' \Vdash \{\rho\}\check{\tau}_1 : u_1$	by $\Gamma \Vdash \tau : u$
$\Gamma' \Vdash \{\rho\}\check{\tau}_1 = \{\rho\}\check{\tau}_1' : u_1$	by $\Gamma \Vdash \tau = \tau' : u$
$\Gamma' \Vdash \{\rho\}\check{\tau}_1' = \{\rho\}\check{\tau}_1 : u_1$	by induction hypothesis (Prop. 4)
$\Gamma' \vdash \{\rho\}\check{\tau}_1' : u_1$	by $\Gamma \Vdash \tau' : u$
$\Gamma' \Vdash s' = s : \{\rho\}\check{\tau}_1$	by induction hypothesis (Prop. 6), conversion
$\Gamma' \Vdash s = s' : \{\rho\}\check{\tau}_1$	by induction hypothesis (Prop. 2), symmetry for terms
$\Gamma' \Vdash \{\rho, s/y\}\tau_2 = \{\rho, s'/y\}\tau_2' : u_2$	by $\Gamma \Vdash \tau = \tau' : u$
$\Gamma' \Vdash s = s : \{\rho\}\check{\tau}_1$	by induction hypothesis (Prop. 1), reflexivity
$\Gamma' \Vdash \{\rho, s/y\}\tau_2 : u_2$	by $\Gamma \Vdash \tau : u$
$\Gamma' \Vdash \{\rho, s'/y\}\tau_2' = \{\rho, s/y\}\tau_2 : u$	by induction hypothesis (Prop. 4), symmetry for types
$\Gamma \Vdash \tau' = \tau : u$	since Γ', ρ, s, s' were arbitrary

Transitivity for Types (Prop. 5): $\boxed{\text{If } \Gamma \Vdash \tau' = \tau'' : u \text{ and } \Gamma \Vdash \tau'' : u \text{ then } \Gamma \Vdash \tau = \tau'' : u.}$

$\Gamma \Vdash (y : \check{\tau}_1') \Rightarrow \tau_1 = \tau' : u$	by assumption
$\Gamma \vdash \tau' \searrow (y : \check{\tau}_2') \Rightarrow \tau_2 : u$	by definition of $\Gamma \Vdash \tau = \tau' : u$ and determinacy of reduction
$\Gamma \vdash \tau'' \searrow (y : \check{\tau}_3') \Rightarrow \tau_3 : u$	by definition of $\Gamma \Vdash \tau' = \tau'' : u$
Assume $\Gamma' \leq_\rho \Gamma$ and $\Gamma' \Vdash s_1 = s_3 : \{\rho\}\check{\tau}_1'$	
$\Gamma' \Vdash \{\rho\}\check{\tau}_1' : u_1$	by $\Gamma \Vdash \tau : u$
$\Gamma' \Vdash s_1 = s_1 : \{\rho\}\check{\tau}_1'$	by induction hypothesis (Prop. 1), reflexivity
$\Gamma' \Vdash \{\rho, s_1/y\}\tau_1 = \{\rho, s_1/y\}\tau_2 : u_2$	by definition of $\Gamma \Vdash \tau = \tau' : u$
$\Gamma' \Vdash \{\rho\}\check{\tau}_1' = \{\rho\}\check{\tau}_2' : u_1$	by definition of $\Gamma \Vdash \tau = \tau' : u$
$\Gamma' \Vdash s_1 = s_1 : \{\rho\}\check{\tau}_2'$	by induction hypothesis (Prop. 6), type conversion
$\Gamma' \Vdash \{\rho, s_1/y\}\tau_2 = \{\rho, s_1/y\}\tau_3 : u_2$	by definition of $\Gamma \Vdash \tau' = \tau'' : u$
$\Gamma' \Vdash \{\rho, s_1/y\}\tau_1 : u_2$	by $\Gamma \Vdash \tau : u$
$\Gamma' \Vdash \{\rho, s_1/y\}\tau_1 = \{\rho, s_1/y\}\tau_3 : u_2$	by induction hypothesis (Prop. 5), transitivity for types
$\Gamma' \Vdash \{\rho\}\check{\tau}_2' = \{\rho\}\check{\tau}_3' : u_1$	by definition of $\Gamma \Vdash \tau' = \tau'' : u$
$\Gamma' \Vdash \{\rho\}\check{\tau}_1' = \{\rho\}\check{\tau}_3' : u_1$	by induction hypothesis (Prop. 5), transitivity for types
$\Gamma' \Vdash s_1 = s_3 : \{\rho\}\tau_3'$	by induction hypothesis (6), type conversion
$\Gamma' \Vdash \{\rho, s_1/y\}\tau_3 = \{\rho, s_3/y\}\tau_3 : u_2$	by $\Gamma \Vdash \tau'' : u$
$\Gamma \Vdash \{\rho, s_1/y\}\tau_1 = \{\rho, s_3/y\}\tau_3 : u_2$	by induction hypothesis (Prop. 5), transitivity for types
$\Gamma \Vdash \tau = \tau'' : u$	since Γ', ρ, s_1, s_3 were arbitrary

Conversion (Prop. 6). $\boxed{\Gamma \Vdash t_1 = t_2 : \tau'.}$

$\Gamma \vdash \tau \equiv \tau' : u$	by Well-formedness Lemma 6.8

$\Gamma \vdash t_1 \searrow w_1 : \tau$ by definition of $\Gamma \Vdash t_1 = t_2 : \tau$
$\Gamma \vdash t_1 \searrow w_1 : \tau'$ by the conversion rule
$\Gamma \vdash t_2 \searrow w_2 : \tau'$ ditto
$\Gamma \vdash \tau' \searrow (y : \breve{\tau}_1') \Rightarrow \tau_2' : u$ by definition of $\Gamma \Vdash \tau = \tau' : u$
Assume $\Gamma' \leq_\rho \Gamma$ and $\Gamma' \Vdash s_1 = s_2 : \{\rho\}\breve{\tau}_1'$
$\Gamma' \Vdash \{\rho\}\breve{\tau}_1 = \{\rho\}\breve{\tau}_1' : u_1$ by definition of $\Gamma \Vdash \tau = \tau' : u_1$
$\Gamma' \Vdash \{\rho\}\breve{\tau}_1 : u_1$ by $\Gamma \Vdash \tau : u$
$\Gamma' \Vdash \{\rho\}\breve{\tau}_1' = \{\rho\}\breve{\tau}_1 : u_1$ by induction hypothesis (Prop. 4), symmetry
$\Gamma' \Vdash s_1 = s_2 : \{\rho\}\breve{\tau}_1$ by induction hypothesis (Prop. 6) on $\Gamma' \Vdash \{\rho\}\breve{\tau}_1' : u_1$, conversion
$\Gamma' \Vdash \{\rho\}w_1\ s_1 = \{\rho\}w_2\ s_2 : \{\rho, s_1/y\}\tau_2$ by assumption $\Gamma \Vdash t_1 = t_2 : \tau$
$\Gamma' \Vdash s_1 = s_1 : \{\rho\}\breve{\tau}_1$ by induction hypothesis (1), reflexivity
$\Gamma' \Vdash \{\rho, s_1/y\}\tau_2 : u_2$ by definition of $\Gamma \Vdash \tau : u$
$\Gamma' \Vdash \{\rho, s_1/y\}\tau_2 = \{\rho, s_1/y\}\tau_2'$ by definition of $\Gamma \Vdash \tau = \tau' : u$
$\Gamma' \Vdash \{\rho\}w_1\ s_1 = \{\rho\}w_2\ s_2 : \{\rho, s_1/y\}\tau_2'$ by induction hypothesis (Prop. 6), conversion
$\Gamma \Vdash t_1 = t_2 : \tau'$ since Γ', ρ, s_1, s_2 were arbitrary

Case. $\boxed{\boxed{\tau = u', \text{ i.e. } \Gamma \Vdash \tau : u \text{ where } \Gamma \vdash \tau \searrow u' : u \text{ and } u' < u}}$

Symmetry for Terms (Prop. 2): To Show: $\boxed{\Gamma \Vdash t_2 = t_1 : u'}$
$\Gamma \Vdash t_2 = t_1 : u'$ by IH using Symmetry for Types (Prop. 4) (since $u' < u$)

Transitivity for Terms (Prop. 3): To Show: $\boxed{\text{If } \Gamma \Vdash t_2 = t_3 : u' \text{ then } \Gamma \Vdash t_1 = t_3 : u'.}$

$\Gamma \vdash t_1 = t_3 : u'$ by IH using Transitivity for Types (Prop. 5) (since $u' < u$)

Symmetry for Types (Prop. 4): To Show: $\boxed{\Gamma \Vdash \tau' = u' : u}$

$\Gamma \vdash \tau' \searrow u' : u$ by $\Gamma \Vdash u' = \tau' : u$
$\Gamma \vdash \tau' = \tau : u$ since $\Gamma \vdash \tau \searrow u' : u$ and $u' < u$ (by assumption), and $\Gamma \vdash \tau' \searrow u' : u$

Transitivity for Types (Prop. 5): To Show: $\boxed{\text{If } \Gamma \Vdash \tau' = \tau'' : u \text{ and } \Gamma \Vdash \tau'' : u \text{ then } \Gamma \Vdash u' = \tau'' : u.}$

$\Gamma \vdash \tau' \searrow u' : u$ $\Gamma \Vdash u' = \tau' : u$
$\Gamma \vdash \tau'' \searrow u' : u$ by $\Gamma \Vdash \tau' = \tau'' : u$
$\Gamma \Vdash u' = \tau'' : u$ using sem. equ. def. and the assumption $u' < u$

Conversion (Prop. 6): To Show: $\boxed{\Gamma \Vdash t_1 = t_2 : \tau'.}$

$\Gamma \Vdash t_1 = t_2 : \tau$ and $\Gamma \Vdash \tau : u$ where $\Gamma \vdash \tau \searrow u' : u$ and $u' < u$ by assumption
$\Gamma \Vdash u' = \tau' : u$ and $u' < u$ by assumption
$\Gamma \vdash \tau' \searrow u' : u$ by $\Gamma \Vdash u' = \tau' : u$
$\Gamma \Vdash t_1 = t_2 : \tau'$ since $\Gamma \Vdash \tau' : u$

Case. $\boxed{\boxed{\tau = x\,\vec{t} \text{ and } \Gamma \vdash \tau \searrow x\,\vec{t} : u \text{ and wne } (x\,\vec{t})}}$

Symmetry for Terms (Prop. 2): To Show: $\boxed{\Gamma \Vdash t_2 = t_1 : x\,\vec{s}.}$

$\Gamma \Vdash t_1 = t_2 : x\,\vec{s}$	by assumption
$\Gamma \vdash t_1 \searrow n_1 : x\,\vec{s}, \Gamma \vdash t_2 \searrow n_2 : x\,\vec{s}, \Gamma \vdash n_1 \equiv n_2 : x\,\vec{s}$, wne n_1, n_2	by $\Gamma \Vdash t_1 = t_2 : x\,\vec{s}$
$\Gamma \vdash n_2 \equiv n_1 : x\,\vec{s}$	by symmetry of $\equiv$
$\Gamma \Vdash t_2 = t_1 : x\,\vec{s}$	by sem. equ. definition

Transitivity for Terms (Prop. 3): To Show $\boxed{\text{If } \Gamma \Vdash t_2 = t_3 : x\,\vec{s} \text{ then } \Gamma \Vdash t_1 = t_3 : x\,\vec{s}.}$

$\Gamma \Vdash t_1 = t_2 : x\,\vec{s}$	by assumption
$\Gamma \vdash t_1 \searrow n_1 : x\,\vec{s}, \Gamma \vdash t_2 \searrow n_2 : x\,\vec{s}, \Gamma \vdash n_1 \equiv n_2 : x\,\vec{s}$, wne n_1, n_2	by $\Gamma \Vdash t_1 = t_2 : x\,\vec{s}$
$\Gamma \vdash t_3 \searrow n_3 : x\,\vec{s}, \Gamma \vdash n_2 \equiv n_3 : x\,\vec{s}$, wne n_3	by $\Gamma \Vdash t_2 = t_3 : x\,\vec{s}$
$\Gamma \vdash n_1 \equiv n_3 : x\,\vec{s}$	by transitivity of $\equiv$
$\Gamma \Vdash t_1 = t_3 : x\,\vec{s}$	by sem. equ. definition

Symmetry for Types (Prop. 4): To Show: $\boxed{\Gamma \Vdash \tau' = x\,\vec{s} : u}$

$\Gamma \Vdash x\,\vec{s} = \tau' : u$ where $\Gamma \vdash \tau \searrow x\,\vec{s} : u$	by assumption
$\Gamma \vdash \tau' \searrow x\,\vec{t} : u$ and $\Gamma \vdash x\,\vec{s} \equiv x\,\vec{t} : u$	
$\Gamma \vdash x\,\vec{t} \equiv x\,\vec{s} : u$	by symmetry of $\equiv$
$\Gamma \Vdash \tau' = \tau : u$	by sem. equ. definition

Transitivity for Types (Prop. 5): To Show: $\boxed{\text{If } \Gamma \Vdash \tau' = \tau'' : u \text{ and } \Gamma \Vdash \tau'' : u \text{ then } \Gamma \Vdash x\,\vec{s} = \tau'' : u.}$

$\Gamma \Vdash x\,\vec{s} = \tau' : u$	by assumption
$\Gamma \vdash \tau' \searrow x\,\vec{s'} : u$ and $\Gamma \vdash x\,\vec{s} \equiv x\,\vec{s'} : u$	by $\Gamma \Vdash x\,\vec{s} = \tau' : u$
$\Gamma \Vdash \tau' = \tau'' : u$	by assumption
$\Gamma \vdash \tau'' \searrow x\,\vec{s''} : u$ and $\Gamma \vdash x\,\vec{s'} \equiv x\,\vec{s''} : u$	by $\Gamma \Vdash \tau' = \tau'' : u$
$\Gamma \vdash x\,\vec{s} = x\,\vec{s''} : u$	by transitivity of $\equiv$
$\Gamma \vdash \tau = \tau'' : u$	by sem. equ. def.

Conversion (Prop. 6): $\boxed{\Gamma \Vdash t_1 = t_2 : \tau'.}$

$\Gamma \Vdash t_1 = t_2 : x\,\vec{s}$	by assumption
$\Gamma \vdash t_1 \searrow n_1 : x\,\vec{s}$ and $\Gamma \vdash t_2 \searrow n_2 : x\,\vec{s}$ and $\Gamma \vdash n_1 \equiv n_2 : x\,\vec{s}$	by $\Gamma \Vdash t_1 = t_2 : x\,\vec{s}$
$\Gamma \Vdash x\,\vec{s} = \tau' : u$	by assumption
$\Gamma \vdash \tau' \searrow x\,\vec{s'} : u$ and $\Gamma \vdash x\,\vec{s} \equiv x\,\vec{s'} : u$	by $\Gamma \Vdash x\,\vec{s} = \tau' : u$
$\Gamma \vdash n_1 \equiv n_2 : x\,\vec{s'}$	using type conversion
$\Gamma \vdash t_1 \searrow n_1 : x\,\vec{s'}$ and $\Gamma \vdash t_2 \searrow n_2 : x\,\vec{s'}$	using type conversion
$\Gamma \Vdash t_1 = t_2 : \tau'$	by sem. equ. def.

□

Finally we establish various elementary properties about our semantic definition that play a key role in the fundamental lemma which we prove later. First, we show that all neutral terms are in the semantic definition.

Lemma 6.11 (Neutral Soundness).
If $\Gamma \Vdash \check{\tau} : u$ *and* $\Gamma \vdash t : \check{\tau}$ *and* $\Gamma \vdash t' : \check{\tau}$ *and* $\Gamma \vdash t \equiv t' : \check{\tau}$ *and wne* t, t' *then* $\Gamma \Vdash t = t' : \check{\tau}$.

Proof. By induction on $\Gamma \Vdash \tau : u$.

Case. $\tau = u$

wne t and wne t'	by assumption
$t = x\,\vec{s}$ and $t' = x\,\vec{s}'$	since wne t and wne t, $\Gamma \vdash t : u$, $\Gamma \vdash t' : u$ and $\Gamma \vdash t \equiv t' : u$
$\Gamma \vdash x\,\vec{s} \equiv x\,\vec{s}' : u$	by assumption $\Gamma \vdash t \equiv t' : u$
$\Gamma \vdash t' \searrow x\,\vec{s}'$ and $\Gamma \vdash t \searrow x\,\vec{s}$	since wne t and wne t'
$\Gamma \vdash x\,\vec{s} = t' : u$	by sem. def.

Case. $\tau = \lceil T \rceil$ where $\Psi \vdash \mathrm{tm}$

$\Gamma \vdash t : \lceil T \rceil$ and $\Gamma \vdash t' : \lceil T \rceil$	by assumption
wne t and wne t'	by assumption
whnf t and whnf t'	by def. of whnf /wne
$t \searrow t$ and $t' \searrow t'$	by def. of $\searrow$
$\Gamma \vdash t \equiv t' : \lceil T \rceil$	by assumption
$\Gamma \vdash t \searrow t : \lceil T \rceil$ and $\Gamma \vdash t' \searrow t' : \lceil T \rceil$	by Def. 4.3
$\Gamma; \Psi \vdash \lfloor t \rfloor_{\mathrm{id}} \searrow_{\mathsf{LF}} \lfloor t \rfloor_{\mathrm{id}} : A$	since wne t
$\Gamma; \Psi \vdash \lfloor t' \rfloor_{\mathrm{id}} \searrow_{\mathsf{LF}} \lfloor t' \rfloor_{\mathrm{id}} : A$	since wne t'
$\Gamma; \Psi \vdash \mathrm{typeof}(\Gamma \vdash t) = \lceil \Phi \vdash \mathrm{tm} \rceil$ and $\Gamma \vdash \Psi \equiv \Phi : \mathrm{ctx}$	by Lemma 5.1
$\Gamma; \Psi \vdash \mathrm{typeof}(\Gamma \vdash t') = \lceil \Phi' \vdash \mathrm{tm} \rceil$ and $\Gamma \vdash \Psi \equiv \Phi' : \mathrm{ctx}$	by Lemma 5.1
$\Gamma \vdash \Phi \equiv \Phi' : \mathrm{ctx}$	by symmetry and transitivity of $\equiv$
$\Gamma; \Psi \Vdash \mathrm{id} = \mathrm{id} : \Psi$	by Lemma 6.5
$\Gamma; \Psi \Vdash \lfloor t \rfloor_{\mathrm{id}} = \lfloor t' \rfloor_{\mathrm{id}} : A$	by sem. def.
$\Gamma \Vdash t = t' : \lceil T \rceil$	by semantic def.

Case. $\tau = (y : \check{\tau}_1) \Rightarrow \tau_2$

$\Gamma \vdash t : (y : \check{\tau}_1) \Rightarrow \tau_2$ and $\Gamma \vdash t' : (y : \check{\tau}_1) \Rightarrow \tau_2$	by assumption
wne t and wne t'	by assumption
whnf t and whnf t'	by def. of whnf /wne
$t \searrow t$ and $t' \searrow t'$	by def. of $\searrow$
$\Gamma \vdash t \equiv t' : (y : \check{\tau}_1) \Rightarrow \tau_2$	by assumption
$\Gamma \vdash t \searrow t : (y : \check{\tau}_1) \Rightarrow \tau_2$	by Def. 4.3
$\Gamma \vdash t' \searrow t' : (y : \check{\tau}_1) \Rightarrow \tau_2$	by Def. 4.3
Assume $\forall \Gamma' \leq_\rho \Gamma.\ \Gamma' \Vdash s = s' : \{\rho\}\check{\tau}_1$	
$\quad\Gamma' \vdash \{\rho\}t \equiv \{\rho\}t' : \{\rho\}((y : \check{\tau}_1) \Rightarrow \tau_2)$	by Weakening Lemma 3.18
$\quad\Gamma' \vdash \{\rho\}t \equiv \{\rho\}t' : (y : \{\rho\}\check{\tau}_1) \Rightarrow \{\rho, y/y\}\tau_2$	by subst. def.
$\quad\Gamma' \vdash s \equiv s' : \{\rho\}\check{\tau}_1$	by Well-formedness Lemma 6.8
$\quad\Gamma' \vdash \{\rho\}t\ s \equiv \{\rho\}t'\ s' : \{\rho, s/y\}\tau_2$	by rule
$\quad$wne $\{\rho\}t$ and wne $\{\rho\}t'$	by Lemma 4.1
$\quad$wne $\{\rho\}t\ s$ and wne $\{\rho\}t'\ s'$	by def. of whnf /wne
$\quad\Gamma' \Vdash \{\rho, s/y\}\tau_2 : u_2$	by $\Gamma \Vdash \tau : u$
$\quad\Gamma' \Vdash \{\rho\}t\ s = \{\rho\}t\ s' : \{\rho, s/y\}\tau_2$	by IH
$\Gamma \Vdash t = t' : (y : \check{\tau}_1) \Rightarrow \tau_2$	by semantic def.

Case. $\tau = x\,\vec{s}$

$\Gamma \vdash \tau \searrow x\ \vec{s} : u$ and wne $(x\ \vec{s})$ by $\Gamma \Vdash \tau : u$
$\Gamma \vdash t \equiv t' : x\ \vec{s}$ and wne t, t' by assumption
$\Gamma \vdash t \searrow t : x\ \vec{s}$ since wne t
$\Gamma \vdash t' \searrow t' : x\ \vec{s}$ since wne t'
$\Gamma \Vdash t = t' : x\ \vec{s}$ by sem. equ. def.

□

We also show that semantic definition are backwards closed.

Lemma 6.12 (Backwards Closure for Computations). *If* $\Gamma \Vdash t_1 = t_2 : \check{\tau}$ *and* $\Gamma \vdash t_1 \searrow w : \check{\tau}$ *and* $\Gamma \vdash t_1' \searrow w : \check{\tau}$ *then* $\Gamma \Vdash t_1' = t_2 : \check{\tau}$.

Proof. We analyse the definition of $\Gamma \Vdash t_1 = t_2 : \check{\tau}$ considering different cases of $\Gamma \Vdash \check{\tau} : u$. □

Lemma 6.13 (Typed Whnf is Backwards Closed). *If* $\Gamma \vdash t \searrow w : (y : \check{\tau}_1) \Rightarrow \tau_2$ *and* $\Gamma \vdash s : \check{\tau}_1$ *and* $\Gamma \vdash w\ s \searrow v : \{s/y\}\tau_2$ *then* $\Gamma \vdash t\ s \searrow v : \{s/y\}\tau_2$.

Proof. Proof by unfolding definition and typing rules and considering different cases for w.
$\Gamma \vdash t : (y : \check{\tau}_1) \Rightarrow \tau_2$ by def. of $\Gamma \vdash t \searrow w : (y : \check{\tau}_1) \Rightarrow \tau_2$
$\Gamma \vdash s : \check{\tau}_1$ by assumption
$\Gamma \vdash t\ s : \{s/y\}\tau_2$ by typing rule
$\Gamma \vdash s \equiv s : \check{\tau}_1$ by reflexivity of $\equiv$
$\Gamma \vdash t \equiv w : (y : \check{\tau}_1) \Rightarrow \tau_2$ by def. of $\Gamma \vdash t \searrow w : (y : \check{\tau}_1) \Rightarrow \tau_2$
$\Gamma \vdash t\ s \equiv w\ s : \{s/y\}\tau_2$ by congruence rules of $\equiv$
$\Gamma \vdash w\ s \equiv v : \{s/y\}\tau_2$ by def. of $\Gamma \vdash w\ s \searrow v : \{s/y\}\tau_2$
$\Gamma \vdash t\ s \equiv v : \{s/y\}\tau_2$ by symmetry and transitivity of $\equiv$
$t \searrow w$ by def. of $\Gamma \vdash t \searrow w : (y : \check{\tau}_1) \Rightarrow \tau_2$
$w\ s \searrow v$ by def. of $\Gamma \vdash w\ s \searrow v : \{s/y\}\tau_2$
$\Gamma \vdash v : \{s/y\}\tau_2$ by def. of $\Gamma \vdash w\ s \searrow v : \{s/y\}\tau_2$
whnf w by definition of $\searrow$

Sub-case: $t \searrow \mathsf{fn}\ x \Rightarrow t'$ and $w = \mathsf{fn}\ x \Rightarrow t'$
$(\mathsf{fn}\ x \Rightarrow t')\ s \searrow v$ where $\{s/x\}t' \searrow v$ by $\Gamma \vdash w\ s \searrow v : \{s/y\}\tau_2$
$t\ s \searrow v$ since $t \searrow \mathsf{fn}\ x \Rightarrow t'$
$\Gamma \vdash t\ s \searrow v : \{s/y\}\tau_2$ by def.

Sub-case: $t \searrow w$ where wne w
$w\ s \searrow w\ s$ since wne $(w\ s)$
$t\ s \searrow w\ s$ by rule
$\Gamma \vdash t\ s \searrow v : \{s/y\}\tau_2$ by def.
□

Lemma 6.14 (Semantic Application). *If* $\Gamma \Vdash t = t' : (y : \check{\tau}_1) \Rightarrow \tau_2$ *and* $\Gamma \Vdash s = s' : \check{\tau}_1$ *then* $\Gamma \Vdash t\ s = t'\ s' : \{s/y\}\tau_2$

Proof. Proof using Well-formedness Lemma 6.8, Backwards closed properties (Lemma 6.13 and 6.12), and Symmetry of semantic equality (Lemma Prop. 2).

$\Gamma \vdash t \searrow w : (y : \check{\tau}_1) \Rightarrow \tau_2$ and
$\Gamma \vdash t' \searrow w' : (y : \check{\tau}_1) \Rightarrow \tau_2$ and
$\forall \Gamma' \leq_\rho \Gamma.\ \Gamma' \Vdash s = s' : \{\rho\}\check{\tau}_1 \Longrightarrow \Gamma' \Vdash (\{\rho\}w)\ s = \{\rho\}w'\ s' : \{\rho, s/y\}\tau_2$ by sem. def.

$\Gamma \Vdash w\ s = w'\ s' : \{s/y\}\tau_2$ choosing Γ for Γ', ρ to be the identity substitution
$\Gamma \vdash w\ s \searrow v : \{s/y\}\tau_2$ by def. of $\Gamma \Vdash w\ s = w'\ s' : \{s/y\}\tau_2$
$\Gamma \vdash s : \tau_1$ and $\Gamma \vdash s' \vdash \tau_1$ by Well-formedness Lemma 6.8
$\Gamma \vdash t\ s \searrow v : \{s/y\}\tau_2$ by Whnf Backwards closed (Lemma 6.13)
$\Gamma \vdash w'\ s' \searrow v : \{s/y\}\tau_2$ by def. of $\Gamma \Vdash w\ s = w'\ s' : \{s/y\}\tau_2$
$\Gamma \vdash t'\ s' \searrow v : \{s/y\}\tau_2$ by Whnf Backwards closed (Lemma 6.13)
$\Gamma \Vdash t\ s = t'\ s' : \{s/y\}\tau_2$ by Semantic Backwards Closure for Computations (Lemma 6.12) and Symmetry (Lemma Prop. 2)

□

7 VALIDITY IN THE MODEL

For normalization, we need to establish that well-typed terms are logically related. However, as we traverse well-typed terms, they do not remain closed. As is customary, we now extend our logical relation to substitutions defining semantic substitutions $\boxed{\Gamma \Vdash \theta = \theta' : \Gamma}$:

$$\frac{\vdash \Gamma'}{\Gamma' \Vdash \cdot = \cdot : \cdot} \qquad \frac{\Gamma' \Vdash \theta = \theta' : \Gamma \quad \Gamma' \Vdash \{\theta\}\check{\tau} = \{\theta'\}\check{\tau} : u \quad \Gamma' \Vdash t = t' : \{\theta\}\check{\tau} \quad \Gamma' \Vdash \{\theta\}\check{\tau} : u}{\Gamma' \Vdash \theta, t/x = \theta', t'/x : \Gamma, x{:}\check{\tau}}$$

Lemma 7.1 (Semantic Weakening of Computation-level Substitutions). *If $\Gamma' \Vdash \theta = \theta' : \Gamma$ and $\Gamma'' \leq_\rho \Gamma'$ then $\Gamma'' \Vdash \{\rho\}\theta = \{\rho\}\theta' : \Gamma$.*

Proof. By induction on $\Gamma' \Vdash \theta = \theta' : \Gamma$ and using semantic weakening lemma 6.2. □

Lemma 7.2 (Semantic Substitution Preserves Equivalence). *Let $\Gamma' \Vdash \theta = \theta' : \Gamma$;*

(1) If $\Gamma; \Psi \vdash M \equiv M : A$ then $\Gamma; \{\theta\}\Psi \vdash \{\theta\}M \equiv \{\theta'\}M : \{\theta\}A$.
(2) If $\Gamma; \Psi \vdash \sigma \equiv \sigma : \Phi$ then $\Gamma; \{\theta\}\Psi \vdash \{\theta\}\sigma \equiv \{\theta'\}\sigma : \{\theta\}\Phi$.
(3) If $\Gamma \vdash t \equiv t : \check{\tau}$ then $\Gamma' \vdash \{\theta\}t \equiv \{\theta'\}t : \{\theta\}\check{\tau}$.

Proof. By induction on M, σ τ, and t. The proof is mostly straightforward; in the case where $t = x$ we know by assumption that $t_x/x \in \theta$ and $t'_x/x \in \theta'$ where $\Gamma' \Vdash t_x = t'_x : \{\theta\}\tau_x$. From Well-formedness of semantic typing (Lemma 6.8), we know that $\Gamma' \vdash t_x \equiv t'_x : \{\theta\}\tau_x$. □

Last, we define validity of computation-level contexts and and computation-level types and terms referring to their semantic definitions (Fig. 21). This allows us to define compactly the fundamental lemma which states that well typed terms correspond to valid terms in our model. Validity here is defined in terms of the semantic definition (Fig. 20). In particular, we say that two terms M and N are equal in our model, if for all computation-level instantiations θ and θ' which are considered semantically equal, we have that $\{\theta\}M$ and $\{\theta'\}N$ are semantically equal.

Lemma 7.3 (Well-formedness of Semantic Substitutions). *If $\Gamma' \Vdash \theta = \theta' : \Gamma$ then $\Gamma' \vdash \theta : \Gamma$ and $\Gamma' \vdash \theta' : \Gamma$ and $\Gamma' \vdash \theta \equiv \theta' : \Gamma$.*

Proof. By induction on $\Gamma' \Vdash \theta = \theta' : \Gamma$. □

Lemma 7.4 (Symmetry and Transitivity of Semantic Substitutions). *Assume $\models \Gamma$.*

(1) If $\Gamma' \Vdash \theta_1 = \theta_2 : \Gamma$ then $\Gamma' \Vdash \theta_2 = \theta_1 : \Gamma$.
(2) If $\Gamma' \Vdash \theta_1 = \theta_2 : \Gamma$ and $\Gamma' \Vdash \theta_2 = \theta_3 : \Gamma$ then $\Gamma' \Vdash \theta_1 = \theta_3 : \Gamma$.

Proof. We prove symmetry by induction on the derivation and transitivity by induction on both derivations using Symmetry, Transitivity, and Conversion for semantic equality (Lemma 6.10); ; reflexivity follows from symmetry and transitivity.

Symmetry: By induction on derivation.

Validity of Context : $\boxed{\models \Gamma}$

$$\frac{}{\models \cdot} \qquad \frac{\models \Gamma \quad \Gamma \models \check{\tau} : u}{\models \Gamma, x : \check{\tau}}$$

Validity of LF Objects : $\boxed{\Gamma; \Psi \models M = N : A}$

$$\frac{\models \Gamma \quad \forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma'; \{\theta\}\Psi \Vdash \{\theta\}M = \{\theta'\}N : \{\theta\}A}{\Gamma; \Psi \models M = N : A}$$

Validity of LF Substitutions : $\boxed{\Gamma; \Psi \models \sigma = \sigma' : \Phi}$

$$\frac{\models \Gamma \quad \forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma'; \{\theta\}\Psi \Vdash \{\theta\}\sigma_1 = \{\theta'\}\sigma' : \{\theta\}\Phi}{\Gamma; \Psi \models \sigma = \sigma' : \Phi}$$

Validity of Types : $\boxed{\Gamma \models \check{\tau} = \check{\tau}' : u}$ and $\boxed{\Gamma \models \check{\tau} : u}$

$$\frac{\models \Gamma \quad \forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}\check{\tau} = \{\theta'\}\check{\tau}' : u}{\Gamma \models \check{\tau} = \check{\tau}' : u} \qquad \frac{\Gamma \models \check{\tau} = \check{\tau} : u}{\Gamma \models \check{\tau} : u}$$

Validity of Terms : $\boxed{\Gamma \models t = t' : \check{\tau}}$ and $\boxed{\Gamma \models t : \check{\tau}}$

$$\frac{\models \Gamma \quad \Gamma \models \check{\tau} : u \quad \forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}t = \{\theta'\}t' : \{\theta\}\check{\tau}}{\Gamma \models t = t' : \check{\tau}} \quad \frac{\Gamma \models t = t : \check{\tau}}{\Gamma \models t : \check{\tau}}$$

Fig. 21. Validity Definition

Case. $\dfrac{}{\Gamma' \Vdash \cdot = \cdot : \cdot}$

$\Gamma' \Vdash \cdot = \cdot : \cdot$ by def.

Case. $\mathcal{D} = \dfrac{\Gamma' \Vdash \theta = \theta' : \Gamma \qquad \Gamma' \Vdash t = t' : \{\theta\}\check{\tau}}{\Gamma' \Vdash \theta, t/x = \theta', t'/x : \Gamma, x{:}\check{\tau}}$

$\Gamma' \Vdash \theta' = \theta : \Gamma$ by IH
$\Gamma' \Vdash t' = t : \{\theta\}\check{\tau}$ by Lemma 6.10 (Symmetry)
$\Gamma' \Vdash \theta', t'/x = \theta, t/x : \Gamma, x{:}\check{\tau}$ by rule

Transitivity: By induction on both derivations.

Case. $\theta_1 = \cdot$, $\theta_2 = \cdot$ and $\theta_3 = \cdot$

$\Gamma' \Vdash \cdot = \cdot : \cdot$ by def.

Case. $\theta_1 = \theta_1', t_1/x$, $\theta_2 = \theta_2', t_2/x$ and $\theta_3 = \theta_3', t_3/x$ and $\Gamma = \Gamma_0, x{:}\check{\tau}$

$\Gamma' \Vdash \theta_1' = \theta_2' : \Gamma_0$ by inversion
$\Gamma' \Vdash \theta_2' = \theta_3' : \Gamma_0$ by inversion
$\Gamma' \Vdash \theta_1' = \theta_3' : \Gamma_0$ by IH
$\Gamma' \Vdash t_1 = t_2 : \{\theta_1'\}\check{\tau}$ by inversion

$\Gamma' \Vdash t_2 = t_3 : \{\theta_2'\}\breve{\tau}$ by inversion
$\Gamma' \Vdash \{\theta_1'\}\breve{\tau} = \{\theta_2'\}\breve{\tau} : u$ by inversion
$\Gamma' \Vdash \{\theta_2'\}\breve{\tau} = \{\theta_3'\}\breve{\tau} : u$ by inversion
$\Gamma' \Vdash \{\theta_2'\}\breve{\tau} = \{\theta_1'\}\breve{\tau} : u$ by Lemma 6.10 (Symmetry)
$\Gamma' \Vdash t_2 = t_3 : \{\theta_1'\}\breve{\tau}$ by Lemma 6.10 (Conversion)
$\Gamma' \Vdash t_1 = t_3 : \{\theta_1'\}\breve{\tau}$ by Lemma 6.10 (Transitivity)
$\Gamma' \Vdash \{\theta_1'\}\breve{\tau} = \{\theta_3'\}\breve{\tau}$ by Lemma 6.10 (Transitivity)
$\Gamma' \Vdash \theta_1 = \theta_3 : \Gamma$ by rule

□

Lemma 7.5 (Context Satisfiability). *If* $\models \Gamma$ *then* $\vdash \Gamma$ *and* $\Gamma \Vdash id(\Gamma) = id(\Gamma) : \Gamma$ *where*

$$\begin{array}{lcl} id(\cdot) & = & \cdot \\ id(\Gamma, x{:}\tau) & = & id(\Gamma), x/x \end{array}$$

Proof. By induction on Γ using Neutral Soundness (Lemma 6.11) and Semantic Weakening (Lemma 7.1). By induction on Γ.

Case. $\Gamma = \cdot$

$\vdash \cdot$ by rules
$\mathsf{id}(\cdot) = \cdot$ by def. of id
$\Gamma' \Vdash \cdot = \cdot : \cdot$ by semantic def.

Case. $\Gamma = \Gamma_0, x{:}\breve{\tau}$

$\models \Gamma_0, x{:}\breve{\tau}$ by assumption
$\models \Gamma_0$ and $\Gamma_0 \models \breve{\tau} : u$ by def. validity
$\vdash \Gamma_0$ and $\Gamma_0 \Vdash \mathsf{id}(\Gamma_0) = \mathsf{id}(\Gamma_0) : \Gamma_0$ by IH
$\forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}\breve{\tau} = \{\theta\}\breve{\tau} : \{\theta\}u$ by def. validity
$\Gamma_0 \Vdash \{\mathsf{id}(\Gamma_0)\}\breve{\tau} = \{\mathsf{id}(\Gamma_0)\}\breve{\tau} : \{\mathsf{id}(\Gamma_0)\}u$ by previous lines
$\Gamma_0 \Vdash \breve{\tau} = \breve{\tau} : u$ by subst. def.
$\Gamma_0 \vdash \breve{\tau} : u$ by semantic typing
$\vdash \Gamma_0, x{:}\breve{\tau}$ by context def.
$\mathsf{wne}\ x$ by def.
$\Gamma_0, x{:}\breve{\tau} \vdash x : \breve{\tau}$ by typing rule
$\Gamma_0, x{:}\breve{\tau} \vdash x \equiv x : \breve{\tau}$ by typing rule
$\Gamma_0, x{:}\breve{\tau} \Vdash x = x : \breve{\tau}$ by Neutral Soundness Lemma 6.11
$\Gamma_0, x{:}\breve{\tau} \Vdash \mathsf{id}(\Gamma_0) = \mathsf{id}(\Gamma_0) : \Gamma_0$ by Sem. Weakening Lemma 7.1
$\Gamma_0, x{:}\breve{\tau} \Vdash \mathsf{id}(\Gamma_0), x/x = \mathsf{id}(\Gamma_0), x/x : \Gamma_0, x{:}\breve{\tau}$ by semantic def.
$\Gamma_0, x{:}\breve{\tau} \Vdash \mathsf{id}(\Gamma_0, x{:}\tau) = \mathsf{id}(\Gamma_0, x{:}\tau) : \Gamma_0, x{:}\breve{\tau}$ by def. of id

□

Lemma 7.6 (Symmetry and Transitivity of Validity).

(1) If $\Gamma \models t = t' : \breve{\tau}$ *then* $\Gamma \models t' = t : \breve{\tau}$.
(2) If $\Gamma \models t_1 = t_2 : \breve{\tau}$ *and* $\Gamma \models t_2 = t_3 : \breve{\tau}$ *then* $\Gamma \models t_1 = t_3 : \breve{\tau}$.

Proof. Using Lemma 7.5 (Context Satisfiability), Lemma 7.4 (Symmetry and Transitivity for Substitutions), Lemma 6.10 (Symmetry and Transitivity for Terms), and Lemma 6.10 (Conversion).

Case. $$\frac{\models \Gamma \qquad \Gamma \models \breve{\tau} : u \qquad \forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}t = \{\theta'\}t' : \{\theta\}\breve{\tau}}{\Gamma \models t = t' : \breve{\tau}}$$

Assume $\Gamma' \Vdash \theta' = \theta : \Gamma$
$\Gamma' \Vdash \theta = \theta' : \Gamma$ by Lemma 7.4 (Symmetry)
$\Gamma' \Vdash \{\theta\}t = \{\theta'\}t' : \{\theta\}\check{\tau}$ by assumption $\Gamma \Vdash t = t' : \check{\tau}$
$\Gamma' \Vdash \{\theta\}t' = \{\theta'\}t : \{\theta\}\check{\tau}$ by Lemma 6.10 (Symmetry)
$\models \Gamma$ by assumption
$\Gamma \Vdash \mathsf{id}(\Gamma) = \mathsf{id}(\Gamma) : \Gamma$ by Lemma 7.5
$\Gamma \models \check{\tau} : u$ by assumption
$\Gamma \models \check{\tau} = \check{\tau} : u$ by def.
$\Gamma' \Vdash \{\theta'\}\check{\tau} = \{\theta\}\check{\tau} : u$ by $\Gamma \models \check{\tau} = \check{\tau} : u$
$\Gamma' \Vdash \{\theta'\}t' = \{\theta\}t : \{\theta'\}\check{\tau}$ by Lemma 6.10 (Conversion)
$\Gamma \models t' = t : \check{\tau}$ by rule

Case. $\dfrac{\models \Gamma \qquad \Gamma \models \check{\tau} : u \quad \forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}t_1 = \{\theta'\}t_2 : \{\theta\}\check{\tau}}{\Gamma \models t_1 = t_2 : \check{\tau}}$ and

Case. $\dfrac{\models \Gamma \qquad \Gamma \models \check{\tau} : u \quad \forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}t_2 = \{\theta'\}t_3 : \{\theta\}\check{\tau}}{\Gamma \models t_2 = t_3 : \check{\tau}}$

Assume $\Gamma' \Vdash \theta = \theta' : \Gamma$
$\Gamma' \Vdash \theta = \theta : \Gamma$ by symmetry and transitivity of substitution (Lemma 7.4)
$\Gamma' \Vdash \{\theta\}t_1 = \{\theta\}t_2 : \{\theta\}\check{\tau}$ by $\Gamma \models t_1 = t_2 : \check{\tau}$
$\Gamma' \Vdash \{\theta\}t_2 = \{\theta'\}t_3 : \{\theta\}\check{\tau}$ by $\Gamma \models t_2 = t_3 : \check{\tau}$
$\Gamma' \Vdash \{\theta\}t_1 = \{\theta'\}t_3 : \{\theta\}\check{\tau}$ by Lemma 6.10 (Transitivity)
$\Gamma \models t_1 = t_3 : \check{\tau}$ by rule
□

Lemma 7.7 (Function Type Injectivity is valid.). *If* $\Gamma \models (y : \check{\tau}_1) \Rightarrow \tau_2 = (y : \check{\tau}_1') \Rightarrow \tau_2' : u_3$ *then* $\Gamma \models \check{\tau}_1 = \check{\tau}_1' : u_1$ *and* $\Gamma, y{:}\check{\tau}_1 \models \tau_2 = \tau_2' : u_2$ *and* $(u_1,\ u_2,\ u_3) \in \Re$

Proof. Proof by unfolding the semantic definitions.

$\Gamma \models (y : \check{\tau}_1) \Rightarrow \tau_2 = (y : \check{\tau}_1') \Rightarrow \tau_2' : u_3$ by assumption
$\models \Gamma$
$\forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}(y : \check{\tau}_1) \Rightarrow \tau_2 = \{\theta'\}(y : \check{\tau}_1') \Rightarrow \tau_2' : \{\theta\}u_3$ by def. of validity

To prove: $\Gamma \models \check{\tau}_1 = \check{\tau}_1' : u_1$
Assume an arbitrary $\Gamma' \Vdash \theta = \theta' : \Gamma$.
$\Gamma' \Vdash \{\theta\}(y : \check{\tau}_1) \Rightarrow \tau_2 = \{\theta'\}(y : \check{\tau}_1') \Rightarrow \tau_2' : u_3$ by previous lines and $\{\theta\}u_3 = u_3$
$\Gamma' \Vdash (y : \{\theta\}\check{\tau}_1) \Rightarrow \{\theta,\ y/y\}\tau_2 = (y : \{\theta'\}\check{\tau}_1') \Rightarrow \{\theta', y/y\}\tau_2' : u_3$ by subst. def.
$(y : \{\theta'\}\check{\tau}_1') \Rightarrow \{\theta',\ y/y\}\tau_2' \searrow (y : \{\theta'\}\check{\tau}_1') \Rightarrow \{\theta',\ y/y\}\tau_2'$ by unfolding semantic def. and $\searrow$
since whnf $((y : \{\theta'\}\check{\tau}_1') \Rightarrow \{\theta',\ y/y\}\tau_2')$

$\forall \Gamma_0 \leq_\rho \Gamma'.\ \Gamma_0 \Vdash \{\rho\}\{\theta\}\check{\tau}_1 = \{\rho\}\{\theta\}\check{\tau}_1' : u_1$ by sem. def.
$\Gamma' \Vdash \{\theta\}\check{\tau}_1 = \{\theta\}\check{\tau}_1' : u_1$ choosing $\Gamma_0 = \Gamma'$ and $\rho = \mathsf{id}(\Gamma')$ and the fact that $\{\mathsf{id}(\Gamma')\}\theta = \theta$
$\forall \Gamma', \theta, \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}\check{\tau}_1 = \{\theta'\}\check{\tau}_1' : u_1$ previous lines
$\Gamma \models \check{\tau}_1 = \check{\tau}_1' : u_1$ def. of validity ($\models$), since Γ', θ, θ' arbitrary

To prove: $\Gamma, y{:}\check{\tau}_1 \models \tau_2 = \tau_2' : u_2$

Assume an arbitrary $\Gamma' \Vdash \theta_2 = \theta_2' : \Gamma, y{:}\check{\tau}_1$.	
$\theta_2 = \theta, s/y$ and $\theta_2' = \theta', s'/y$	
$\Gamma' \Vdash \theta = \theta' : \Gamma$ and $\Gamma' \Vdash s = s' : \{\theta\}\check{\tau}_1$	by inversion on $\Gamma' \Vdash \theta_2 = \theta_2' : \Gamma, y{:}\check{\tau}_1$
$\forall \Gamma_0 \leq_\rho \Gamma'.\ \Gamma_0 \Vdash s = s' : \{\rho\}\{\theta\}\check{\tau}_1 \Longrightarrow \Gamma' \Vdash \{\rho, s/y\}\{\theta, y/y\}\tau_2 = \{\rho, s'/y\}\{\theta', y/y\}\tau_2'$	by sem. def.
$\Gamma' \Vdash s = s' : \{\theta\}\check{\tau}_1 \Longrightarrow \Gamma' \Vdash \{s/y\}\{\theta, y/y\}\tau_2 = \{s'/y\}\{\theta', y/y\}\tau_2'$	by choosing $\rho = \mathrm{id}(\Gamma')$
$\Gamma' \Vdash s = s' : \{\theta\}\check{\tau}_1 \Longrightarrow \Gamma' \Vdash \{\theta, s/y\}\tau_2 = \{\theta', s'/y\}\tau_2'$	by subst. def.
$\Gamma' \Vdash \{\theta, s/y\}\tau_2 = \{\theta', s'/y\}\tau_2'$	by previous line
$\Gamma' \Vdash \{\theta_2\}\tau_2 = \{\theta_2'\}\tau_2'$	by previous lines
$\Gamma, y{:}\check{\tau}_1 \models \tau_2 = \tau_2' : u_2$	by def. of validity, since Γ', θ, θ' arbitrary

□

Theorem 7.8 (Completeness of Validity). *If $\Gamma \models t = t' : \check{\tau}$ then $\Gamma \vdash t : \check{\tau}$ and $\Gamma \vdash t' : \check{\tau}$ and $\Gamma \vdash t \equiv t' : \check{\tau}$ and $\Gamma \vdash \check{\tau} : u$.*

Proof. Unfolding of validity definition relying on context satisfiability (Lemma 7.5) and Well-Formedness of Semantic Typing (Lemma 6.8).

$\Gamma \models t = t' : \check{\tau}$	by assumption
$\models \Gamma$	by validity definition
$\vdash \Gamma$ and $\Gamma \Vdash \mathrm{id}(\Gamma) = \mathrm{id}(\Gamma) : \Gamma$	by Lemma 7.5
$\forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}t = \{\theta\}t' : \{\theta\}\check{\tau}$	by validity definition
$\Gamma \Vdash \{\mathrm{id}(\Gamma)\}t = \{\mathrm{id}(\Gamma)\}t' : \{\mathrm{id}(\Gamma)\}\check{\tau}$	by previous lines
$\Gamma \Vdash t = t' : \check{\tau}$	by subst. def.
$\Gamma \vdash t' : \check{\tau}, \Gamma \vdash t : \check{\tau}, \Gamma \vdash t \equiv t' : \check{\tau}$ and $\Gamma \vdash \check{\tau} : u$	by Well-Formedness of Seman. Typ. (Lemma 6.8)

□

The fundamental lemma (Lemma 7.13) states that well-typed terms are valid. The proof proceeds by mutual induction on the typing derivation for LF-objects and computations. To structure the proof of the fundamental lemma that well-typed computations are valid, we consider the validity of type conversion, computation-level functions, applications, and recursion individually.

Lemma 7.9 (Validity of Type Conversion). *If $\Gamma \models \check{\tau} = \check{\tau}' : u$ and $\Gamma \models t : \check{\tau}$ then $\Gamma \models t : \check{\tau}'$.*

Proof. By definition relying on semantic type conversion lemma (Lemma 6.10 (6)).

$\Gamma \models t : \check{\tau}$	by assumption
$\Gamma \models t = t : \check{\tau}$	by validity def.
Assume $\Gamma' \Vdash \theta = \theta' : \Gamma$	
$\Gamma' \Vdash \{\theta\}t = \{\theta'\}t : \{\theta\}\check{\tau}$	by validity def. $\Gamma \models t = t : \check{\tau}$
$\Gamma \Vdash \{\theta\}\check{\tau} = \{\theta'\}\check{\tau}' : u$	by validity def. $\Gamma \models \check{\tau} = \check{\tau}' : u$
$\Gamma \Vdash \{\theta\}t = \{\theta'\} : \{\theta\}\check{\tau}'$	by Lemma 6.10 (Conversion)
$\Gamma \models t = t : \check{\tau}'$	since Γ', θ, θ' arbitrary

□

Lemma 7.10 (Validity of Functions). *If $\Gamma, y{:}\check{\tau}_1 \models t : \tau_2$ then $\Gamma \models fn\ y \Rightarrow t : (y : \check{\tau}_1) \Rightarrow \tau_2$.*

Proof. We unfold the validity definitions, relying on completeness of validity (Lemma 7.8), semantic weakening of computation-level substitutions (Lemma 7.1), Well-formedness Lemma 6.8, Backwards Closure Lemma 6.12, Symmetry property of semantic equality (Lemma 6.10).

$\Gamma, y{:}\check{\tau}_1 \models t : \tau_2$	by assumption
$\Gamma, y : \check{\tau}_1 \models t = t : \tau_2$	by def. validity
$\forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma, y{:}\check{\tau}_1 \Longrightarrow \Gamma' \Vdash \{\theta\}t = \{\theta'\}t : \{\theta\}\tau_2$	by def. of validity

$\Gamma, y : \check{\tau}_1 \models \tau_2 : u$ by def. of validity
$\Gamma, y : \check{\tau}_1 \models \tau_2 = \tau_2 : u$ by inversion on validity
$\models \Gamma, y : \check{\tau}_1$ by inversion on validity
$\models \Gamma$ by inversion on validity
$\Gamma \models \check{\tau}_1 : u$ by inversion on validity
$\Gamma \models \check{\tau}_1 = \check{\tau}_1 : u$ by inversion on validity

TO SHOW:

(1) $\models \Gamma$
(2) $\Gamma \models (y : \check{\tau}_1) \Rightarrow \tau_2 : u$, i.e.
$\forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2) = \{\theta'\}((y : \check{\tau}_1) \Rightarrow \tau_2) : u$
(3) $\forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}(\mathsf{fn}\ y \Rightarrow t) = \{\theta'\}(\mathsf{fn}\ y \Rightarrow t) : \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$

(1) SHOW: $\models \Gamma$
$\models \Gamma, y{:}\check{\tau}_1$ by assumption $\Gamma, y : \check{\tau}_1 \models t = t : \tau_2$
$\models \Gamma$ by inversion on $\models \Gamma, y{:}\check{\tau}_1$

(2) SHOW: $\forall \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2) = \{\theta'\}((y : \check{\tau}_1) \Rightarrow \tau_2) : u$
Assume $\Gamma' \Vdash \theta = \theta' : \Gamma$

(2.a) SHOW: $\forall \Gamma'' \leq_\rho \Gamma'.\Gamma'' \Vdash s = s' : \{\rho\}\{\theta\}\check{\tau}_1 \Longrightarrow \Gamma'' \Vdash \{\{\rho\}\theta, s/y\}\tau_2 = \{\{\rho\}\theta', s'/y\}\tau_2 : u$
Assume $\Gamma'' \leq_\rho \Gamma'.\Gamma'' \Vdash s = s' : \{\rho\}\{\theta\}\check{\tau}_1$
$\Gamma'' \Vdash \{\rho\}\theta = \{\rho\}\theta' : \Gamma$ by weakening sem. subst. (Lemma 7.1)
$\Gamma'' \Vdash \{\rho\}\theta, s/y = \{\rho\}\theta', s'/y : \Gamma, y{:}\check{\tau}_1$ by sem. def.
$\Gamma'' \Vdash \{\{\rho\}\theta, s/y\}\tau_2 = \{\{\rho\}\theta', s'/y\}\tau_2 : u$ by def. of $\Gamma, y : \check{\tau}_1 \models \tau_2 = \tau_2 : u$

(2.b) SHOW: $\forall \Gamma'' \leq_\rho \Gamma'.\Gamma'' \Vdash \{\rho\}\{\theta\}\check{\tau}_1 = \{\rho\}\{\theta'\}\check{\tau}_1 : u$

$\Gamma'' \Vdash \{\rho\}\theta = \{\rho\}\theta' : \Gamma$ by weakening sem. subst. (Lemma 7.1)
$\Gamma'' \Vdash \{\rho\}\{\theta\}\check{\tau}_1 = \{\rho\}\{\theta'\}\check{\tau}_1 : u$ by def. of $\Gamma \models \check{\tau}_1 = \check{\tau}_1 : u$

(2.c) SHOW: $\Gamma' \vdash \{\theta'\}((y : \check{\tau}_1) \Rightarrow \tau_2) \searrow \{\theta'\}((y : \check{\tau}_1) \Rightarrow \tau_2) : u$
$\Gamma \vdash (y : \check{\tau}_1) \Rightarrow \tau_2 : u$ by Completeness of Validity (Lemma 7.8)
(using validity of functions which we show under (3))
$\Gamma' \vdash \theta : \Gamma$ and $\Gamma' \vdash \theta' : \Gamma$ by Well-formedness of semantic subst. (Lemma 7.3)
$\Gamma' \vdash \{\theta'\}((y : \check{\tau}_1) \Rightarrow \tau_2) : u$ by computation-level subst. lemma (Lemma 3.11)
$\mathsf{whnf}\ ((y : \{\theta'\}\check{\tau}_1) \Rightarrow \{\theta', y/y\}\tau_2)$ by def. and subst. prop.

(3) SHOW: $\forall \Gamma',\ \theta,\ \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}(\mathsf{fn}\ y \Rightarrow t) = \{\theta'\}(\mathsf{fn}\ y \Rightarrow t) : \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$
Assume $\Gamma' \Vdash \theta = \theta' : \Gamma$; Showing: $\Gamma' \Vdash \{\theta\}(\mathsf{fn}\ y \Rightarrow t) = \{\theta'\}(\mathsf{fn}\ y \Rightarrow t) : \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$

(3.a) SHOW: $\Gamma' \vdash \{\theta\}(\mathsf{fn}\ y \Rightarrow t) \searrow w : \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$
and $\Gamma' \vdash \{\theta'\}(\mathsf{fn}\ y \Rightarrow t) \searrow w' : \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$
$\mathsf{fn}\ y \Rightarrow \{\theta, y/y\}t \searrow w$ where $w = \mathsf{fn}\ y \Rightarrow \{\theta, y/y\}t$ since $\mathsf{whnf}\ (\mathsf{fn}\ y \Rightarrow \{\theta, y/y\}t)$
$\mathsf{fn}\ y \Rightarrow \{\theta', y/y\}t \searrow w'$ where $w' = \mathsf{fn}\ y \Rightarrow \{\theta', y/y\}t$ since $\mathsf{whnf}\ (\mathsf{fn}\ y \Rightarrow \{\theta', y/y\}t)$
$\Gamma, y{:}\check{\tau}_1 \vdash t : \tau_2$ by Well-formedness of semantic typing (Lemma 6.8)
$\Gamma' \vdash \theta : \Gamma$ and $\Gamma' \vdash \theta' : \Gamma$ by Well-formedness of semantic subst. (Lemma 7.3)
$\Gamma', y{:}\{\theta\}\check{\tau}_1 \vdash \theta, y/y : \Gamma, y{:}\check{\tau}_1$ and $\Gamma', y{:}\{\theta'\}\check{\tau}_1 \vdash \theta', y/y : \Gamma, y{:}\check{\tau}_1'$ by comp. subst.
$\Gamma', y{:}\{\theta\}\tau_1 \vdash \{\theta, y/y\}t : \{\theta, y/y\}\tau_2$ and

$\Gamma', y{:}\{\theta'\}\tau_1 \vdash \{\theta', y/y\}t : \{\theta', y/y\}\tau_2$ by computation-level subst. lemma (Lemma 3.11)
$\Gamma' \vdash \{\theta\}(\mathsf{fn}\ y \Rightarrow t) : \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$ and
$\Gamma' \vdash \{\theta'\}(\mathsf{fn}\ y \Rightarrow t) : \{\theta'\}((y : \check{\tau}_1) \Rightarrow \tau_2)$ by typing rule
$\Gamma' \vdash \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2) = \{\theta'\}((y : \check{\tau}_1) \Rightarrow \tau_2) : u$ by (2)
$\Gamma' \vdash \{\theta'\}(\mathsf{fn}\ y \Rightarrow t) : \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$ Conversion (Lemma 6.10(6))

(3.b) SHOW: $\forall \Gamma'' \leq_\rho \Gamma'.\Gamma'' \Vdash s = s' : \{\rho\}\{\theta\}\check{\tau}_1 \Longrightarrow \Gamma'' \Vdash \{\rho\}w\ s = \{\rho\}w'\ s' : \{\rho, s/y\}\{\theta, y/y\}\tau_2$
Assume $\Gamma'' \leq_\rho \Gamma'.\Gamma'' \Vdash s = s' : \{\rho\}\{\theta\}\check{\tau}_1$
$\Gamma'' \Vdash \{\rho\}\theta = \{\rho\}\theta' : \Gamma$ by weakening sem. subst. (Lemma 7.1)
$\Gamma'' \Vdash \{\rho\}\theta, s/y = \{\rho\}\theta', s'/y : \Gamma, y{:}\check{\tau}_1$ by sem. def.
$\Gamma'' \Vdash \{\{\rho\}\theta, s/y\}t = \{\{\rho\}\theta', s'/y\}t : \{\{\rho\}\theta, s/y\}\tau_2$ using $\Gamma, y : \check{\tau}_1 \models t = t : \tau_2$
$\Gamma'' \Vdash \{\{\rho\}\theta, s/y\}t \searrow v : \{\{\rho\}\theta, s/y\}\tau_2$
$\Gamma'' \Vdash \{\{\rho\}\theta', s'/y\}t \searrow v' : \{\{\rho\}\theta, s/y\}\tau_2$ sem. definition $\Vdash$
$\Gamma'' \Vdash \{\rho\}\{\theta\}\mathsf{fn}\ y \Rightarrow t \searrow \{\rho\}\{\theta\}\mathsf{fn}\ y \Rightarrow t : \{\rho\}\{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$ since whnf $(\{\rho\}\{\theta\}\mathsf{fn}\ y \Rightarrow t)$
and $\Gamma'' \vdash \mathsf{fn}\ y \Rightarrow \{\{\rho\}\theta, y/y\}t : \{\rho\}\{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$
$\Gamma'' \Vdash (\mathsf{fn}\ y \Rightarrow \{\{\rho\}\theta, y/y\}t)\ s \searrow v : \{\{\rho\}\theta, s/y\}\tau_2$ by rules (typing and $\searrow$)
$\Gamma'' \Vdash (\mathsf{fn}\ y \Rightarrow \{\{\rho\}\theta, y/y\}t)\ s = \{\{\rho\}\theta', s'/y\}t : \{\{\rho\}\theta, s/y\}\tau_2$ by Backwards Closed (Lemma 6.12)
$\Gamma'' \Vdash \{\{\rho\}\theta, s/y\}\tau_2 = \{\{\rho\}\theta', s'/y\}\tau_2 : u$ by $\Gamma, y : \check{\tau}_1 \models \tau_2 = \tau_2 : u$
$\Gamma'' \Vdash \{\{\rho\}\theta', s'/y\}t = (\mathsf{fn}\ y \Rightarrow \{\{\rho\}\theta, y/y\}t)\ s : \{\{\rho\}\theta, s/y\}\tau_2$ Symmetry (Lemma 6.10(2))
$\Gamma'' \Vdash \{\rho\}\{\theta'\}\mathsf{fn}\ y \Rightarrow t \searrow \{\rho\}\{\theta'\}\mathsf{fn}\ y \Rightarrow t : \{\rho\}\{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$ since whnf $(\{\rho\}\{\theta'\}\mathsf{fn}\ y \Rightarrow t)$
and $\Gamma'' \vdash \mathsf{fn}\ y \Rightarrow \{\{\rho\}\theta', y/y\}t : \{\rho\}\{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$
$\Gamma'' \Vdash (\mathsf{fn}\ y \Rightarrow \{\{\rho\}\theta', y/y\}t)\ s' \searrow v' : \{\{\rho\}\theta, s/y\}\tau_2$ by rules (typing and $\searrow$)
$\Gamma'' \Vdash (\mathsf{fn}\ y \Rightarrow \{\{\rho\}\theta', y/y\}t)\ s' = (\mathsf{fn}\ y \Rightarrow \{\{\rho\}\theta, y/y\}t)\ s : \{\rho\{\theta\}, s/y\}\tau_2$
by Backwards Closed (Lemma 6.12)
$\Gamma'' \Vdash (\mathsf{fn}\ y \Rightarrow \{\{\rho\}\theta, y/y\}t)\ s = (\mathsf{fn}\ y \Rightarrow \{\{\rho\}\theta', y/y\}t)\ s' : \{\rho\{\theta\}, s/y\}\tau_2$
by Symmetry (Lemma 6.10(2))
$\Gamma'' \Vdash (\{\rho\}\mathsf{fn}\ y \Rightarrow \{\theta, y/y\}t)\ s = (\{\rho\}\mathsf{fn}\ y \Rightarrow \{\theta', y/y\}t)\ s' : \{\rho, s/y\}\{\theta, y/y\}\tau_2$ by comp. subst.
$\Gamma'' \Vdash \{\rho\}w\ s = \{\rho\}w'\ s' : \{\rho, s/y\}\{\theta, y/y\}\tau_2$ since $w = \mathsf{fn}\ y \Rightarrow \{\theta, y/y\}t$
and $w' = \mathsf{fn}\ y \Rightarrow \{\theta', y/y\}t$
□

Lemma 7.11 (Validity of Recursion).
Let $\Gamma \vdash rec^{\mathcal{I}}\,(b_v \mid b_{\mathsf{app}} \mid b_{\mathtt{lam}}\,)\ \Psi\ t : \{\Psi/\psi, t/m\}\tau$ *and* $\mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (m : \lceil\psi \vdash \mathsf{tm}\rceil) \Rightarrow \tau$.
If $\Gamma \models \mathcal{I} : u$ *and* $\Gamma \models t : \lceil\Psi \vdash \mathsf{tm}\rceil$ *and*
$\Gamma, \psi : \mathsf{tm\_ctx}, p : \lceil\psi \vdash_\# \mathsf{tm}\rceil \models t_v : \{p/y\}\tau$ *and*
$\Gamma, \psi : \mathsf{tm\_ctx}, m : \lceil\psi \vdash \mathsf{tm}\rceil, n : \lceil\psi \vdash \mathsf{tm}\rceil, f_m : \{m/y\}\tau, f_n : \{n/y\}\tau \models t_{\mathsf{app}} : \{\lceil\psi \vdash \mathsf{app}\ \lfloor m\rfloor_{id}\ \lfloor n\rfloor_{id}\rceil/y\}\tau$ *and*
$\Gamma, \psi : \mathsf{tm\_ctx}, m : \lceil\psi, x : \mathsf{tm} \vdash \mathsf{tm}\rceil, f_m : \{\lceil\psi, x : \mathsf{tm}\rceil/\psi, m/y\}\tau \models t_{\mathtt{lam}} : \{\lceil\psi \vdash \mathtt{lam}\ \lambda x.\lfloor m\rfloor_{id}\,\rceil/y\}\tau$
then $\Gamma \models rec^{\mathcal{I}}\,(b_v \mid b_{\mathsf{app}} \mid b_{\mathtt{lam}}\,)\ \Psi\ t : \{\Psi/\psi, t/m\}\tau$.

Proof. We assume $\Gamma' \Vdash \theta = \theta' : \Gamma$, and show $\Gamma' \Vdash \{\theta\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t) = \{\theta'\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t) : \{\theta\}\{\Psi/\psi,\ t/m\}\tau$ by considering different cases for $\Gamma' \Vdash \{\theta\}t = \{\theta'\}t : \lceil\{\theta\}\Psi \vdash \mathsf{tm}\rceil$. Let $\Psi' = \{\theta\}\Psi$. In the case where $\Gamma' \vdash \{\theta\}t \searrow \lceil\hat{\Psi}' \vdash M\rceil : \lceil\Psi' \vdash \mathsf{tm}\rceil$ and $\Gamma' \vdash \{\theta'\}t \searrow \lceil\Psi' \vdash N\rceil : \lceil\{\theta\}\Psi \vdash \mathsf{tm}\rceil$ and $\Gamma'; \Psi' \Vdash M = N : \mathsf{tm}$, we proceed by an inner induction on $\Gamma'; \Psi' \Vdash M = N : \mathsf{tm}$ exploiting on Back. Closed Lemma (6.12).

We now give the full proof.

Let $\mathrm{rec}^{\mathcal{I}}\ \mathcal{B} = \mathrm{rec}^{\mathcal{I}}\,(b_v \mid b_{\mathsf{app}} \mid b_{\mathtt{lam}}\,)$

Assume $\Gamma' \Vdash \theta = \theta' : \Gamma$;
TO SHOW: $\quad \Gamma' \Vdash \{\theta\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t) = \{\theta'\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t) : \{\theta\}\{\Psi/\psi,\ t/m\}\tau$

$\Gamma' \Vdash \{\theta\}t = \{\theta'\}t : \lceil\{\theta\}\Psi \vdash \mathsf{tm}\rceil$ by validity of $\Gamma \models t : \lceil\Psi \vdash \mathsf{tm}\rceil$

Let $\Psi' = \{\theta\}\Psi$. We now proceed to prove:
Case. $\Gamma' \vdash \{\theta\}t \searrow w : \lceil\Psi' \vdash \mathsf{tm}\rceil$
and $\Gamma' \vdash \{\theta'\}t \searrow w' : \lceil\Psi' \vdash \mathsf{tm}\rceil$
and $\Gamma';\Psi' \Vdash \lfloor w\rfloor_{\mathsf{id}} = \lfloor w'\rfloor_{\mathsf{id}} : \mathsf{tm}$

We write M for $\lfloor w\rfloor_{\mathsf{id}}$ and N for $\lfloor w'\rfloor_{\mathsf{id}}$ below.

If $\Gamma';\Psi' \Vdash M = N : \mathsf{tm}$
then $\Gamma' \Vdash \{\theta\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \lceil\Psi\rceil)\ \lceil\hat{\Psi} \vdash M\rceil = \{\theta'\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \lceil\Psi\rceil)\ \lceil\hat{\Psi} \vdash N\rceil : \{\theta,\ \Psi'/\psi, \lceil\hat{\Psi} \vdash M\rceil/m\}\tau$

by induction on M, i.e. we may appeal to the induction hypothesis if the term M has made progress and has stepped using $\searrow_{\mathsf{LF}}$ and hence is "smaller"

Sub-case: $\Gamma';\Psi' \vdash M \searrow_{\mathsf{LF}} \mathsf{app}\ M_1\ M_2 : \mathsf{tm}$ and $\Gamma';\Psi' \vdash N \searrow_{\mathsf{LF}} \mathsf{app}\ N_1\ N_2 : \mathsf{tm}$
$\Gamma';\Psi' \Vdash M_1 = N_1 : \mathsf{tm}$ and $\Gamma';\Psi' \Vdash M_2 = N_2 : \mathsf{tm}$

$\Gamma' \Vdash \{\theta\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \lceil\Psi\rceil)\ \lceil\hat{\Psi} \vdash M_1\rceil = \{\theta'\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \lceil\Psi\rceil)\ \lceil\hat{\Psi} \vdash N_1\rceil : \{\theta,\ \Psi'/\psi, \lceil\hat{\Psi} \vdash M_1/m\rceil\}\tau$ by IH(i)
$\Gamma' \Vdash \{\theta\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \lceil\Psi\rceil)\ \lceil\hat{\Psi}' \vdash M_2\rceil = \{\theta'\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \lceil\Psi'\rceil)\ \lceil\hat{\Psi}' \vdash N_2\rceil : \{\theta,\ \Psi'/\psi, \lceil\hat{\Psi} \vdash M_2\rceil/m\}\tau$ by IH(i)
$\Gamma' \vdash \{\theta\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ \lceil\hat{\Psi} \vdash M_i\rceil : \{\theta,\ \Psi'/\psi, \lceil\hat{\Psi} \vdash M_i\rceil/m\}\tau$ by Well-form. of Sem. Def.
$\Gamma' \vdash \{\theta\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ \lceil\hat{\Psi} \vdash N_i\rceil : \{\theta',\ \Psi'/\psi, \lceil\hat{\Psi} \vdash N_i\rceil/m\}\tau$ by Well-form. of Sem. Def. and Type Conv.
let $\theta_{\mathsf{app}} = \theta, \Psi'/\psi, \lceil\hat{\Psi} \vdash M_1\rceil/m,\ \lceil\hat{\Psi} \vdash M_2\rceil/n$
$\{\theta\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \lceil\Psi\rceil)\ \lceil\hat{\Psi} \vdash M_1\rceil/f_m, \{\theta\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ \lceil\hat{\Psi} \vdash M_2\rceil/f_n$
let $\theta'_{\mathsf{app}} = \theta', \Psi'/\psi, \lceil\hat{\Psi} \vdash N_1\rceil/m,\ \lceil\hat{\Psi} \vdash N_2\rceil/n$
$\{\theta'\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ \lceil\hat{\Psi} \vdash N_1\rceil/f_m, \{\theta'\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ \lceil\hat{\Psi} \vdash N_2\rceil/f_n$
let $\Gamma_{\mathsf{app}} = \Gamma, \psi : \lceil\mathsf{tm\_ctx}\rceil, m : \lceil\psi \vdash \mathsf{tm}\rceil, n : \lceil\psi \vdash \mathsf{tm}\rceil, f_m : \{m/y\}\tau, f_n : \{n/y\}\tau$
$\Gamma' \Vdash \theta_{\mathsf{app}} = \theta'_{\mathsf{app}} : \Gamma_{\mathsf{app}}$
$\Gamma' \Vdash \{\theta_{\mathsf{app}}\}t_{\mathsf{app}} = \{\theta'_{\mathsf{app}}\}t_{\mathsf{app}} : \{\theta_{\mathsf{app}}\}\{\lceil\psi \vdash \mathsf{app}\ \lfloor m\rfloor_{\mathsf{id}}\ \lfloor n\rfloor_{\mathsf{id}}\rceil/y\}\tau$ by sem. def. of t_{app}
$\Gamma' \Vdash \{\theta_{\mathsf{app}}\}t_{\mathsf{app}} = \{\theta'_{\mathsf{app}}\}t_{\mathsf{app}} : \{\theta, \Psi'/\psi,\ \lceil\hat{\Psi} \vdash \mathsf{app}\ M_1\ M_2\rceil/m\}\tau$ by subst. def.
$\Gamma' \Vdash C = C : (\{\theta\}\Psi \vdash \mathsf{tm})$ by reflexivity (Lemma 6.7) where $C = \hat{\Psi} \vdash \mathsf{app}\ M_1\ M_2$
$\Gamma' \Vdash \{\theta\}t = \lceil\hat{\Psi} \vdash \mathsf{app}\ M_1\ M_2\rceil : \Psi' \vdash \mathsf{tm}$ since $\Gamma' \vdash \{\theta\}t \searrow \lceil C\rceil : (\{\theta\}\Psi \vdash \mathsf{tm})$
$\Gamma' \Vdash \{\theta, \Psi'/\psi,\ \{\theta\}t/m\}\tau = \{\theta,\ \Psi'/\psi, \lceil\hat{\Psi} \vdash \mathsf{app}\ M_1\ M_2\rceil/m\}\tau : u$ by sem. def. of $\Gamma \Vdash \mathcal{I} : u$
$\Gamma' \Vdash \{\theta_{\mathsf{app}}\}t_{\mathsf{app}} = \{\theta'_{\mathsf{app}}\}t_{\mathsf{app}} : \{\theta, \Psi'/\psi,\ \{\theta\}t/m\}\tau$ by type conversion
$\Gamma' \vdash \{\theta_{\mathsf{app}}\}t_{\mathsf{app}} \searrow v : \{\theta, \Psi'/\psi,\ \{\theta\}t/m\}\tau$ by previous sem. def.
$\Gamma';\Psi' \vdash \{\theta\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ @\ \Psi\ t) \searrow v : \{\theta, \Psi'/\psi,\ \{\theta\}t/m\}\tau$
since $\Gamma' \vdash \{\theta\}t \searrow \lceil\hat{\Psi} \vdash M\rceil : \mathsf{tm}$ and $\Gamma';\Psi' \vdash M \searrow_{\mathsf{LF}} \mathsf{app}\ M_1\ M_2 : \mathsf{tm}$
$\Gamma' \Vdash \{\theta\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ @\ \Psi\ t) = \{\theta'_{\mathsf{app}}\}t_{\mathsf{app}} : \{\theta, \Psi'/\psi,\ \{\theta\}t/m\}\tau$ by Back. Closed (Lemma 6.12)
$\Gamma' \vdash \{\theta'_{\mathsf{app}}\}t_{\mathsf{app}} \searrow v' : \{\theta, \Psi'/\psi,\ \{\theta\}t/m\}\tau$ by previous sem. def.
$\Gamma';\Psi' \vdash \{\theta'\}(\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ @\ \Psi\ t) \searrow v' : \{\theta, \Psi'/\psi,\ \{\theta\}t/m\}\tau$ (using type conversion)
since $\Gamma' \vdash \{\theta'\}t \searrow \lceil\hat{\Psi} \vdash N\rceil : \mathsf{tm}$ and $\Gamma';\Psi' \vdash N \searrow_{\mathsf{LF}} \mathsf{app}\ N_1\ N_2 : \mathsf{tm}$

$\Gamma' \Vdash \{\theta\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ @\ \Psi\ t) = \{\theta'\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ @\ \Psi\ t) : \{\theta, \Psi'/\psi,\ \{\theta\}t/m\}\tau$ by Back. Closed (Lemma 6.12)
$\Gamma \models \text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t : \mathcal{I}$ since Γ', θ, θ' were arbitrary

Sub-Case. Other Sub-Cases where $\Gamma'; \Psi' \vdash \{\theta\}M \searrow_{\text{LF}} \texttt{lam}\ \lambda x.M : \text{tm}$ and $\Gamma'; \Psi' \vdash \{\theta\}M \searrow_{\text{LF}} x : \text{tm}$ where $x : \text{tm} \in \Psi'$ are similar.

Sub-Case. $\Gamma'; \Psi' \vdash \{\theta\}M \searrow_{\text{LF}} r_1 : \text{tm}$ where $r_1 = \lfloor t_1 \rfloor_{\sigma_1}$ and wne t_1
and $\Gamma'; \Psi' \vdash \{\theta'\}M \searrow_{\text{LF}} r_2 : \text{tm}$ where $r_2 = \lfloor t_2 \rfloor_{\sigma_2}$ and wne t_2
$\Gamma'; \Psi' \Vdash \sigma_1 = \sigma_2 : \Phi$ since $\Gamma'; \Psi' \Vdash_{LF} \{\theta\}M = \{\theta'\}M : \text{tm}$
$\Gamma'; \Psi' \vdash \sigma_1 \equiv \sigma_2 : \Phi$ by Well-formedness Lemma
$\Gamma' \vdash t_1 \equiv t_2 : \lceil \Phi \vdash \text{tm} \rceil$ since $\Gamma'; \Psi' \Vdash_{LF} \{\theta\}M = \{\theta'\}M : \text{tm}$
$\Gamma'; \Psi' \vdash r_1 \equiv r_2 : \text{tm}$
$\Gamma' \vdash \{\theta\}\Psi \equiv \{\theta'\}\Psi : \text{tm\_ctx}$ Sem. Subst. Preserve Equiv. (Lemma 7.2)
since $\Gamma' \vdash \theta \equiv \theta' : \Gamma$
$\Gamma' \vdash \{\theta\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ \lceil \hat{\Psi} \vdash r_1 \rceil \equiv \{\theta'\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ \lceil \hat{\Psi} \vdash r_2 \rceil : \{\theta, \Psi'/\psi,\ \{\theta\}t/m\}\tau$ by $\equiv$
$\Gamma' \vdash \{\theta\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ \{\theta\}t \searrow \{\theta\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ \lceil \hat{\Psi} \vdash r_1 \rceil : \{\theta, \Psi'/\psi,\ \{\theta\}t/m\}\tau$ by $\searrow$
$\Gamma' \vdash \{\theta'\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ \{\theta'\}t \searrow \{\theta'\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ \lceil \hat{\Psi} \vdash r_2 \rceil : \{\theta, \Psi'/\psi,\ \{\theta\}t/m\}\tau$ by $\searrow$ and type conversion
$\Gamma' \Vdash \{\theta\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ @\ \Psi\ t) = \{\theta'\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ @\ \Psi\ t) : \{\theta, \Psi'/\psi,\ \{\theta\}t/m\}\tau$ Back. Closed (Lemma 6.12)
and Symmetry

Case. wne w and wne w' and $\Gamma' \vdash w \equiv w' : \lceil \Psi' \vdash \text{tm} \rceil$

wne $\{\theta\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ @\ \Psi)\ w$ and wne $\{\theta'\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ @\ \Psi)\ w'$ by def. of wne
$\Gamma' \vdash \{\theta\}\text{rec}^{\mathcal{I}}\ \mathcal{B} \equiv \{\theta'\}\text{rec}^{\mathcal{I}}\ \mathcal{B}$ (short for all the branches remain equivalent) Sem. Subst. Preserve Equiv. (Lemma 7.2)
$\Gamma' \vdash \{\theta\}\Psi \equiv \{\theta'\}\Psi : \text{tm\_ctx}$ Sem. Subst. Preserve Equiv. (Lemma 7.2)
$\Gamma' \Vdash \{\theta\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ @\ \Psi)\ w \equiv \{\theta'\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ @\ \Psi)\ w' : \{\theta, \Psi'/\psi,\ \{\theta\}t/m\}\tau$ by $\equiv$
$\Gamma' \vdash \{\theta\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ t \searrow \{\theta\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ w : \{\theta, \Psi/\psi,\ \{\theta\}t/m\}\tau$ by type conversion and $\searrow$ rule
$\Gamma' \vdash \{\theta\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ t \searrow \{\theta\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi)\ w' : \{\theta, \Psi/\psi,\ \{\theta\}t/m\}\tau$ by type conversion and $\searrow$ rule
$\Gamma' \Vdash \{\theta\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t) = \{\theta'\}(\text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t) : \{\theta, \Psi/\psi,\ \{\theta\}t/m\}\tau$ by Back. Closed Lemma (twice) (6.12)
and Symmetry
$\Gamma \models \text{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ t : \mathcal{I}$ since Γ', θ, θ' were arbitrary

□

Lemma 7.12 (Validity of Application). *If* $\Gamma \models t : (y : \check{\tau}_1) \Rightarrow \tau_2$ *and* $\Gamma \models s : \check{\tau}_1$ *then* $\Gamma \models t\ s : \{s/y\}\tau_2$.

Proof. By definition relying on semantic type application lemma (Lemma 6.14). □

Theorem 7.13 (Fundamental Theorem).

(1) If $\vdash \Gamma$ *then* $\models \Gamma$.
(2) If $\Gamma; \Psi \vdash M : A$ *then* $\Gamma; \Psi \models M = M : A$.
(3) If $\Gamma; \Psi \vdash \sigma : \Phi$ *then* $\Gamma; \Psi \models \sigma = \sigma : \Phi$.
(4) If $\Gamma; \Psi \vdash M \equiv N : A$ *then* $\Gamma; \Psi \models M = N : A$.
(5) If $\Gamma; \Psi \vdash \sigma \equiv \sigma' : \Phi$ *then* $\Gamma; \Psi \models \sigma = \sigma' : \Phi$.
(6) If $\Gamma \vdash t : \tau$ *then* $\Gamma \models t : \tau$.
(7) If $\Gamma \vdash t \equiv t' : \tau$ *then* $\Gamma \models t = t' : \tau$.

Proof. By induction on the first derivation using the previous lemma on validity of application (7.12), Backwards Closed (6.12), Well-formedness Lemma 6.8, Lemma 7.9, Lemma 3.9, Lemma 7.10, Sem. weakening lemma 6.2, Validity of

Recursion Lemma 7.11.

$\boxed{\text{If } \Gamma \vdash t : \tau \text{ then } \Gamma \models t : \tau.}$

Case. $\mathcal{D} = \dfrac{\Gamma \vdash t : \lceil \Phi \vdash A \rceil \quad \Gamma; \Psi \vdash \sigma : \Phi}{\Gamma; \Psi \vdash \lfloor t \rfloor_\sigma : [\sigma]A}$

$\Gamma \models t : \lceil \Phi \vdash A \rceil$ by IH
$\Gamma; \Psi \models \sigma = \sigma : \Phi$ by IH

Assume $\Gamma' \Vdash \theta = \theta' : \Gamma$
$\Gamma' \Vdash \{\theta\}t = \{\theta'\}t : \{\theta\}\lceil \Phi \vdash A \rceil$ by def. $\Gamma \models t : \lceil \Phi \vdash A \rceil$

Sub-case: $\Gamma' \vdash \{\theta\}t \searrow \lceil C \rceil : \{\theta\}\lceil \Phi \vdash A \rceil$ and $\Gamma' \vdash \{\theta'\}t \searrow \lceil C' \rceil : \{\theta\}\lceil \Phi \vdash A \rceil$
and $\Gamma' \Vdash C = C' : \{\theta\}(\Phi \vdash A)$

$C = \hat{\Phi} \vdash M$ and $C' = \hat{\Phi} \vdash N$ by inversion on $\Gamma' \Vdash C = C' : \{\theta\}(\Phi \vdash A)$
$\Gamma'; \{\theta\}\Psi \Vdash \{\theta\}\sigma = \{\theta'\}\sigma : \{\theta\}\Phi$ by $\Gamma; \Psi \models \sigma : \Phi$
$\Gamma'; \{\theta\}\Phi \Vdash M = N : \{\theta\}A$ by def. of $\Gamma' \Vdash C = C' : \{\theta\}(\Phi \vdash A)$
$\Gamma'; \{\theta\}\Psi \Vdash [\{\theta\}\sigma]M = [\{\theta'\}\sigma]N : \{\theta\}[\sigma]A$ by Lemma 6.4
$\Gamma'; \{\theta\}\Psi \vdash [\{\theta\}\sigma]M \searrow_{\mathsf{LF}} W : \{\theta\}[\sigma]A$ by previous line
$\Gamma' \vdash \{\theta\}t \searrow \lceil C \rceil : \{\theta\}\lceil \Phi \vdash A \rceil$ by restating the condition of the case we are in
$\Gamma'; \{\theta\}\Psi \vdash \{\theta\}(\lfloor t \rfloor_\sigma) \searrow_{\mathsf{LF}} W : \{\theta\}[\sigma]A$ by whnf rules
$\Gamma'; \{\theta\}\Psi \vdash [\{\theta'\}\sigma]N \searrow_{\mathsf{LF}} W' : \{\theta\}[\sigma]A$ by previous line
$\Gamma' \vdash \{\theta'\}t \searrow \lceil C' \rceil : \{\theta\}\lceil \Phi \vdash A \rceil$ by restating the condition of the case we are in
$\Gamma'; \{\theta\}\Psi \vdash \{\theta'\}(\lfloor t \rfloor_\sigma) \searrow_{\mathsf{LF}} W' : \{\theta\}[\sigma]A$ by whnf rules
$\Gamma';; \{\theta\}\Psi \Vdash \{\theta\}(\lfloor t \rfloor_\sigma) = \{\theta'\}(\lfloor t \rfloor_\sigma) : \{\theta\}[\sigma]A$ by Backwards Closed Lemma 6.12 (twice) and symmetry.

$\Gamma; \Psi \models \lfloor t \rfloor_\sigma : [\sigma]A$ by abstraction, since Γ', θ, θ' were arbitrary

Sub-case: $\Gamma' \vdash \{\theta\}t \searrow w : \{\theta\}\lceil \Phi \vdash A \rceil$ and $\Gamma' \vdash \{\theta'\}t \searrow w' : \{\theta\}\lceil \Phi \vdash A \rceil$
and wne w and wne w' and $\Gamma \vdash w \equiv w' : \{\theta\}\lceil \Phi \vdash A \rceil$

$\Gamma' \vdash \theta(\lfloor t \rfloor_\sigma) \searrow \lfloor w \rfloor_{\{\theta\}\sigma}$ by whnf rules
$\Gamma' \vdash \theta'(\lfloor t \rfloor_\sigma) \searrow \lfloor w' \rfloor_{\{\theta\}\sigma}$ by whnf rules
$\Gamma'\{\theta\}\Psi \Vdash \{\theta\}\sigma = \{\theta'\}\sigma : \{\theta\}\Phi$ by def. $\Gamma; \Psi \models \sigma = \sigma : \Phi$
$\Gamma'; \{\theta\}\Psi \vdash \{\theta\}\sigma \equiv \{\theta'\}\sigma : \{\theta\}\Phi$ by Well-formedness Lemma 6.8
$\Gamma'; \{\theta\}\Psi \vdash \{\theta\}(\lfloor t \rfloor_\sigma) \equiv \{\theta'\}(\lfloor t \rfloor_\sigma) : \{\theta\}[\sigma]A$ by $\equiv$ rules
$\Gamma' \Vdash \{\theta\}\Phi = \{\theta\}\Phi : \mathsf{ctx}$ Reflexivity
$\mathsf{typeof}(\Gamma' \vdash w) = \lceil \{\theta\}\Phi \vdash \mathsf{tm} \rceil$ since $\Gamma' \vdash w : \{\theta\}\lceil \Phi \vdash A \rceil$
$\mathsf{typeof}(\Gamma' \vdash w') = \lceil \{\theta\}\Phi \vdash \mathsf{tm} \rceil$ since $\Gamma' \vdash w' : \{\theta\}\lceil \Phi \vdash A \rceil$
$\Gamma'; \{\theta\}\Psi \Vdash \{\theta\}(\lfloor t \rfloor_\sigma) = \{\theta'\}(\lfloor t \rfloor_\sigma) : \{\theta\}[\sigma]A$ by semantic def.
$\Gamma; \Psi \models \lfloor t \rfloor_\sigma : [\sigma]A$ by abstraction, since Γ', θ, θ' were arbitrary

Case. $\mathcal{D} = \dfrac{y : \check{\tau} \in \Gamma}{\Gamma \vdash y : \check{\tau}}$

$\mathcal{C} : \vdash \Gamma$ and $\mathcal{C} < \mathcal{D}$ by Lemma 3.9
Assume $\Gamma', \theta, \theta'.\ \Gamma' \Vdash \theta = \theta' : \Gamma$
$\Gamma' \Vdash t = s : \{\theta_i\}\check{\tau}$ by sem. def. of $\Gamma' \Vdash \theta = \theta' : \Gamma$

$\Gamma' \Vdash \{\theta\}y = \{\theta'\}y : \{\theta\}\check{\tau}$ subst. def. where $\{\theta\}y = t$ and $\{\theta'\}y = s$
$\Gamma \models y = y : \check{\tau}$ by def. of validity

Case. $\mathcal{D} = \dfrac{\Gamma \vdash t : \tau' \qquad \Gamma \vdash \tau' \equiv \tau : u}{\Gamma \vdash t : \tau}$

$\Gamma \models t : \tau'$ by IH
$\Gamma \models \tau = \tau' : u$ by IH
$\Gamma \models t : \tau$ by Lemma 7.9

Case. $\mathcal{D} = \dfrac{\Gamma \vdash C : T}{\Gamma \vdash \lceil C \rceil : \lceil T \rceil}$

$\mathcal{C} : \vdash \Gamma$ and $\mathcal{C} < \mathcal{D}$ by Lemma 3.9
$\models \Gamma$ by IH
Let $C = \hat{\Psi} \vdash M$ and $T = \Psi \vdash A$.
$\Gamma; \Psi \vdash M : A$ by inversion on $\Gamma \vdash C : T$
$\Gamma; \Psi \models M : A$ by IH
Assume $\Gamma', \theta, \theta'. \Gamma' \Vdash \theta = \theta'$
$\Gamma; \{\theta\}\Psi \Vdash \{\theta\}M = \{\theta'\}M : \{\theta\}A$ using $\Gamma; \Psi \models M : A$
$\Gamma \Vdash \{\theta\}(\hat{\Psi} \vdash M) = \{\theta'\}(\hat{\Psi} \vdash M) : \{\theta\}T$ by sem. def.
$\Gamma \Vdash \{\theta\}\lceil C \rceil = \{\theta'\}\lceil C \rceil : \{\theta\}\lceil T \rceil$ by previous line
$\Gamma \models \lceil C \rceil = \lceil C \rceil : \lceil T \rceil$ by abstraction, since Γ', θ, θ' were arbitrary
$\Gamma \models \lceil C \rceil : \lceil T \rceil$ by def. of validity

Case. $\mathcal{D} = \dfrac{\Gamma \vdash t : (y : \check{\tau}_1) \Rightarrow \tau_2 \qquad \Gamma \vdash s : \check{\tau}_1}{\Gamma \vdash t\ s : \{s/y\}\tau_2}$

$\Gamma \models s : \check{\tau}_1$ by IH
$\Gamma \models t : (y : \check{\tau}_1) \Rightarrow \tau_2$ by IH
$\Gamma \models t\ s : \{s/y\}\tau_2$ by Lemma 7.12

Case. $\mathcal{D} = \dfrac{\Gamma, y : \check{\tau}_1 \vdash t : \tau_2}{\Gamma \vdash \mathsf{fn}\ y \Rightarrow t : (y : \check{\tau}_1) \Rightarrow \tau_2}$

$\mathcal{C} : \vdash \Gamma, y{:}\check{\tau}_1$ and $\mathcal{C} < \mathcal{D}$ by Lemma 3.9
$\models \Gamma, y{:}\check{\tau}_1$ by IH
$\models \Gamma$ by def. of validity
$\Gamma, y : \check{\tau}_1 \models t : \tau_2$ by IH
$\Gamma \models (\mathsf{fn}\ y \Rightarrow t) : (y : \check{\tau}_1) \Rightarrow \tau_2$ Lemma 7.10

Case. $\mathcal{D} = \dfrac{\vdash \Gamma}{\Gamma \vdash u_1 : u_2}\ (u_1, u_2) \in \mathfrak{A}$

$\Gamma \vdash u_1 \searrow u_1 : u_2$ by rules and typing assumption
$u_1 \leq u_2$ by $(u_1, u_2) \in \mathfrak{A}$

Case. $\mathcal{D} = \dfrac{\Gamma \vdash \check{\tau}_1 : u_1 \qquad \Gamma, y{:}\check{\tau}_1 \vdash \tau_2 : u_2}{\Gamma \vdash (y : \check{\tau}_1) \Rightarrow \tau_2 : u_3}\ (u_1,\ u_2,\ u_3) \in \mathfrak{R}$

$\Gamma \models \check{\tau}_1 : u_1$ by IH

$\forall \Gamma'.\ \Gamma' \Vdash \theta = \theta' : \Gamma \Longrightarrow \Gamma' \Vdash \{\theta\}\check{\tau}_1 = \{\theta'\}\check{\tau}_1 : u_1$ by sem. def.

$\Gamma, y{:}\check{\tau}_1 \models \tau_2 : u_2$ by IH

$\forall\ \Gamma' \Vdash \theta = \theta' : \Gamma, y{:}\check{\tau}_1 \Longrightarrow \Gamma' \Vdash \{\theta\}\tau_2 = \{\theta'\}\tau_2 : u_1$ by sem. def.

Assume $\Gamma' \Vdash \theta = \theta' : \Gamma$

$\Gamma' \Vdash \{\theta\}\check{\tau}_1 = \{\theta'\}\check{\tau}_1 : u_1$ by previous lines

Assume $\Gamma'' \leq_\rho \Gamma'$

$\Gamma'' \Vdash \{\rho\}\{\theta\}\check{\tau}_1 = \{\rho\}\{\theta'\}\check{\tau}_1 : u_1$ sem. weakening for computations lemma 6.9

$\forall \Gamma''.\ \Gamma'' \leq_\rho \Gamma'.\Gamma'' \Vdash \{\rho\}\{\theta\}\check{\tau}_1 = \{\rho\}\{\theta'\}\check{\tau}_1 : u_1$ by previous lines

Assume $\Gamma'' \leq_\rho \Gamma'$ and $\Gamma'' \Vdash t = t' : \{\{\rho\}\theta\}\check{\tau}_1$

$\Gamma'' \Vdash \{\rho\}\theta = \{\rho\}\theta' : \Gamma$ by sem. weakening lemma 7.1

$\Gamma'' \Vdash \{\rho\}\theta, t/y = \rho\{\theta'\}, t'/y : \Gamma, y{:}\check{\tau}_1$ by sem. def.

$\Gamma'' \Vdash \{\{\rho\}\theta, t/y\}\tau_2 = \{\rho\{\theta'\}, t'/y\}\tau_2 : u_2$ by $\Gamma, y{:}\check{\tau}_1 \models \tau_2 : u_2$
choosing $\theta = \{\rho\}\theta, t/y$ and $\theta' = \rho\{\theta'\}, t'/y$

$\Gamma'' \Vdash \{\rho, t/y\}\{\theta, y/y\}\tau_2 = \{\rho, t'/y\}\{\theta', y/y\}\tau_2 : u_2$ by subst. def.

$\forall \Gamma''.\ \Gamma'' \leq_\rho \Gamma'.\ \Gamma'' \Vdash t = t' : \{\{\rho\}\theta\}\check{\tau}_1 \Longrightarrow \Gamma'' \Vdash \{\rho, t/y\}\{\theta, y/y\}\tau_2 = \{\rho, t'/y\}\{\theta', y/y\}\tau_2 : u_2$
by previous lines

$\Gamma' \Vdash \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2 : u_3) = \{\theta'\}((y : \check{\tau}_1) \Rightarrow \tau_2 : u_2)$ by sem. def.

$\Gamma \models ((y : \check{\tau}_1) \Rightarrow \tau_2 : u_3) = ((y : \check{\tau}_1) \Rightarrow \tau_2 : u_3)$ by sem. def.

$\Gamma \models (y : \check{\tau}_1) \Rightarrow \tau_2 : u_3$ by def. of validity

Case. $\mathcal{D} = \dfrac{\Gamma \vdash T}{\Gamma \vdash \lceil T \rceil : u}$

$\models \Gamma$

Assume $\Gamma' \Vdash \theta = \theta' : \Gamma$

$\Gamma \models_{\mathsf{LF}} T = T$ by IH

$\Gamma' \Vdash_{\mathsf{LF}} \{\theta\}T = \{\theta'\}T$ since $\Gamma \models_{\mathsf{LF}} T = T$

$\Gamma' \vdash \lceil\{\theta\}T\rceil \searrow \lceil\{\theta\}T\rceil : u$ since whnf $\lceil\{\theta\}T\rceil$

$\Gamma' \Vdash \{\theta\}\lceil T\rceil = \{\theta'\}\lceil T\rceil : u$ by sem. def.

$\Gamma \models \lceil T\rceil = \lceil T\rceil : \mathsf{U}_k$ since Γ', θ, θ' are arbitrary

$\Gamma \models \lceil T\rceil : \mathsf{U}_k$ by def. of validity

Case. $\mathcal{D} = \dfrac{\Gamma \vdash t : \lceil \Psi \vdash \mathsf{tm}\rceil \quad \Gamma \vdash \mathcal{I} : u \quad \Gamma \vdash b_v : \mathcal{I} \quad \Gamma \vdash b_{\mathsf{app}} : \mathcal{I} \quad \Gamma \vdash b_{\mathtt{lam}} : \mathcal{I}}{\Gamma \vdash \mathsf{rec}^{\mathcal{I}}\,(b_v \mid b_{\mathsf{app}} \mid b_{\mathtt{lam}}\,)\ \Psi\ t : \{\Psi/\psi,\ t/y\}\tau}$

where $\mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (y : \lceil \psi \vdash \mathsf{tm}\rceil) \Rightarrow \tau$

$\Gamma \models \mathcal{I} : u$ by IH

$\Gamma \models t : \lceil \Psi \vdash \mathsf{tm}\rceil$ by IH

$\Gamma, \psi : \mathsf{tm\_ctx}, p : \lceil\, \psi \vdash_\# \mathsf{tm}\rceil \vdash t_v : \{p/y\}\tau$ by typing inversion

$\Gamma, \psi : \mathsf{tm\_ctx}, m : \lceil\, \psi \vdash \mathsf{tm}\rceil, n : \lceil\, \psi \vdash \mathsf{tm}\rceil f_m : \{m/y\}\tau, f_n : \{n/y\}\tau$
$\vdash t_{\mathsf{app}} : \{\lceil\, \psi \vdash \mathsf{app}\ \lfloor m\rfloor_{\mathsf{id}}\ \lfloor n\rfloor_{\mathsf{id}}\rceil/y\}\tau$ by typing inversion

$\Gamma, \phi : \mathsf{tm\_ctx}, m : \lceil\, \phi, x : \mathsf{tm} \vdash \mathsf{tm}\rceil, f_m : \{\lceil\, \phi, x : \mathsf{tm}\rceil/\psi, m/y\}\tau$
$\vdash t_{\mathtt{lam}}\ : \{\phi/\psi, \lceil\, \phi \vdash \mathtt{lam}\ \lambda x.\lfloor m\rfloor_{\mathsf{id}}\ \rceil/y\}\tau$ by typing inversion

$C : \vdash \Gamma, \psi : \mathsf{tm\_ctx}, p : \lceil\, \psi \vdash_\# \mathsf{tm}\rceil$ and $C < \mathcal{D}$ by Lemma 3.9

$\models \Gamma$ by def. of validity

$\Gamma, \psi : \mathsf{tm\_ctx}, p : \lceil \psi \vdash_{\#} \mathsf{tm} \rceil \models t_v : \{p/y\}\tau$ by IH

$\Gamma, \psi : \mathsf{tm\_ctx}, m : \lceil \psi \vdash \mathsf{tm} \rceil, n : \lceil \psi \vdash \mathsf{tm} \rceil f_m : \{m/y\}\tau, f_n : \{n/y\}\tau$

$\models t_{\mathsf{app}} : \{\lceil \psi \vdash \mathsf{app}\ \lfloor m \rfloor_{\mathsf{id}}\ \lfloor n \rfloor_{\mathsf{id}} \rceil / y\}\tau$ by IH

$\Gamma, \phi : \mathsf{tm\_ctx}, m : \lceil \phi, x : \mathsf{tm} \vdash \mathsf{tm} \rceil, f_m : \{\lceil \phi, x : \mathsf{tm} \rceil / \psi, m/y\}\tau$

$\models t_{\mathtt{lam}} : \{\phi/\psi, \lceil \phi \vdash \mathtt{lam}\ \lambda x. \lfloor m \rfloor_{\mathsf{id}} \rceil / y\}\tau$ by IH

$\Gamma \models \mathsf{rec}^{\mathcal{I}}\,(b_v \mid b_{\mathsf{app}} \mid b_{\mathtt{lam}})\ \Psi\ t : \mathcal{I}$ by Validity of Recursion Lemma 7.11

$\boxed{\text{If } \Gamma \vdash t \equiv t' : \tau \text{ then } \Gamma \models t = t' : \tau.}$

Case. $\mathcal{D} = \dfrac{\Gamma \vdash \mathsf{fn}\ y \Rightarrow t : (y{:}\check{\tau}_1) \Rightarrow \tau_2 \qquad \Gamma \vdash s : \check{\tau}_1}{\Gamma \vdash (\mathsf{fn}\ y \Rightarrow t)\ s \equiv \{s/y\}t : \{s/y\}\tau_2}$

To Show: $\Gamma \models (\mathsf{fn}\ y \Rightarrow t)\ s = \{s/y\}t : \{s/y\}\tau_2$

$\mathcal{C} : \vdash \Gamma$ and $\mathcal{C} < \mathcal{D}$ by Lemma 3.9

$\models \Gamma$ by IH

Assume $\Gamma' \Vdash \theta = \theta' : \Gamma$

$\Gamma \models \mathsf{fn}\ y \Rightarrow t : (y : \check{\tau}_1) \Rightarrow \tau_2$ by IH

$\Gamma \models s : \check{\tau}_1$ by IH

$\Gamma' \Vdash \{\theta\}s = \{\theta'\}s : \{\theta\}\check{\tau}_1$ by $\Gamma \models s : \check{\tau}_1$

$\Gamma \Vdash \{\theta\}(\mathsf{fn}\ y \Rightarrow t) = \{\theta'\}(\mathsf{fn}\ y \Rightarrow t) : \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$ by previous lines (def. of $\models$)

$\Gamma \vdash \{\theta\}(\mathsf{fn}\ y \Rightarrow t) \searrow \{\theta\}(\mathsf{fn}\ y \Rightarrow t) : \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$ by sem. equ. def.

$\Gamma \vdash \{\theta'\}(\mathsf{fn}\ y \Rightarrow t) \searrow \{\theta'\}(\mathsf{fn}\ y \Rightarrow t) : \{\theta\}((y : \check{\tau}_1) \Rightarrow \tau_2)$ by sem. equ. def.

$\Gamma \Vdash \{\theta\}(\mathsf{fn}\ y \Rightarrow t)\ \{\theta\}s = \{\theta'\}(\mathsf{fn}\ y \Rightarrow t)\ \{\theta'\}s' : \{\theta, \{\theta\}s/y\}\tau_2$ by sem. equ. def.

$\Gamma \vdash \{\theta\}((\mathsf{fn}\ y \Rightarrow t)\ s) \searrow w : \{\theta, s/y\}\tau_2$ by sem. equ. def

$\Gamma \vdash \{\theta, \{\theta\}s/y\}t \searrow w : \{\theta, s/y\}\tau_2$ by inversion on $\searrow$

$\Gamma \vdash \{\theta'\}((\mathsf{fn}\ y \Rightarrow t)\ s) \searrow w' : \{\theta, s/y\}\tau_2$ by sem. equ. def

$\Gamma \vdash \{\theta', \{\theta'\}s/y\}t \searrow w' : \{\theta, s/y\}\tau_2$ by inversion on $\searrow$

$\Gamma \Vdash (\mathsf{fn}\ y \Rightarrow t)\ s = \{s/y\}t : \{s/y\}\tau_2$ by Backwards Closure (Lemma 6.12)

Case. $\mathcal{D} = \dfrac{\Gamma \vdash t : \lceil \Psi \vdash A \rceil}{\Gamma \vdash \lceil \hat{\Psi} \vdash \lfloor t \rfloor_{\mathsf{wk}_{\hat{\Psi}}} \rceil \equiv t : \lceil \Psi \vdash A \rceil}$

To Show: $\Gamma \models \lceil \hat{\Psi} \vdash \lfloor t \rfloor_{\mathsf{wk}_{\hat{\Psi}}} \rceil = t : \lceil \Psi \vdash A \rceil$

Assume $\Gamma' \Vdash \theta = \theta' : \Gamma$

$\Gamma \models t : \lceil \Psi \vdash A \rceil$ by IH

$\Gamma' \Vdash \{\theta\}t = \{\theta'\}t : \{\theta\}\lceil \Psi \vdash A \rceil$ by $\Gamma \models t : \lceil \Psi \vdash A \rceil$

$\Gamma' \vdash \{\theta\}t \searrow w : \{\theta\}\lceil \Psi \vdash A \rceil$ by sem. def.

$\Gamma' \vdash \{\theta'\}t \searrow w' : \{\theta\}\lceil \Psi \vdash A \rceil$ by sem. def.

$\Gamma'; \{\theta\}\Psi \Vdash \lfloor w \rfloor_{\mathsf{id}} = \lfloor w' \rfloor_{\mathsf{id}} : \{\theta\}A$ by sem. def.

We note that w is either n where wne n or $\lceil \hat{\Psi} \vdash M \rceil$

Sub-case w is neutral, i.e. wne w

$\Gamma'; \{\theta\}\Psi \vdash \lfloor \{\theta\}t \rfloor_{\mathsf{wk}_{\hat{\Psi}}} \searrow_{\mathsf{LF}} \lfloor w \rfloor_{\mathsf{id}} : \{\theta\}A$ since $\Gamma' \vdash \{\theta\}t \searrow w : \{\theta\}\lceil \Psi \vdash A \rceil$

$\Gamma'; \{\theta\}\Psi \vdash \lfloor \{\theta\}\lceil \hat{\Psi} \vdash \lfloor t \rfloor_{\mathsf{wk}_{\hat{\Psi}}} \rceil \rfloor_{\mathsf{id}} \searrow_{\mathsf{LF}} \lfloor w \rfloor_{\mathsf{id}} : \{\theta\}A$ by $\searrow_{\mathsf{LF}}$

$\Gamma; \{\theta\}\Psi \Vdash \lfloor \{\theta\}\lceil \hat{\Psi} \vdash \lfloor t \rfloor_{\mathsf{wk}_{\hat{\Psi}}} \rceil \rfloor_{\mathsf{id}} = \lfloor w' \rfloor_{\mathsf{id}} : \{\theta\}A$

using $\Gamma'; \{\theta\}\Psi \Vdash \lfloor w \rfloor_{\mathsf{id}} = \lfloor w' \rfloor_{\mathsf{id}} : \{\theta\}A$ and Backwards Closure (Lemma 6.12)

$\Gamma; \{\theta\}\Psi \vdash \{\theta\}\lceil\hat{\Psi} \vdash \lfloor t \rfloor_{\mathsf{wk}_{\hat{\Psi}}}\rceil \searrow \{\theta\}\lceil\hat{\Psi} \vdash \lfloor t \rfloor_{\mathsf{wk}_{\hat{\Psi}}}\rceil : \{\theta\}\lceil\Psi \vdash A\rceil$ since whnf $\{\theta\}\lceil\hat{\Psi} \vdash \lfloor t \rfloor_{\mathsf{wk}_{\hat{\Psi}}}\rceil$

$\Gamma' \Vdash \{\theta\}\lceil\hat{\Psi} \vdash \lfloor t \rfloor_{\mathsf{wk}_{\hat{\Psi}}}\rceil = \{\theta'\}t : \{\theta\}\lceil\Psi \vdash A\rceil$ by sem. def.

Sub-case $w = \lceil\hat{\Psi} \vdash M\rceil$

$\Gamma'; \{\theta\}\Psi \vdash \lfloor\lceil\hat{\Psi} \vdash M\rceil\rfloor_{\mathsf{id}} \searrow_{\mathsf{LF}} N : \{\theta\}A$ by $\Gamma'; \{\theta\}\Psi \Vdash \lfloor w \rfloor_{\mathsf{id}} = \lfloor w' \rfloor_{\mathsf{id}} : \{\theta\}A$

$\Gamma'; \{\theta\}\Psi \vdash M \searrow_{\mathsf{LF}} N : \{\theta\}A$ by $\searrow_{\mathsf{LF}}$

$\Gamma'; \{\theta\}\Psi \vdash \lfloor\{\theta\}\lceil\hat{\Psi} \vdash \lfloor t \rfloor_{\mathsf{wk}_{\hat{\Psi}}}\rceil\rfloor_{\mathsf{id}} \searrow_{\mathsf{LF}} N : \{\theta\}A$

$\Gamma'; \{\theta\}\Psi \Vdash N = \lfloor w' \rfloor_{\mathsf{id}} : \{\theta\}A$ by sem. equ.

$\Gamma'; \{\theta\}\Psi \Vdash \lfloor\{\theta\}\lceil\hat{\Psi} \vdash \lfloor t \rfloor_{\mathsf{wk}_{\hat{\Psi}}}\rceil\rfloor_{\mathsf{id}} = \lfloor w' \rfloor_{\mathsf{id}} : \{\theta\}A$ by Backwards Closure (Lemma 6.12)

$\Gamma' \Vdash \{\theta\}\lceil\hat{\Psi} \vdash \lfloor t \rfloor_{\mathsf{wk}_{\hat{\Psi}}}\rceil = \{\theta'\}t : \{\theta\}\lceil\Psi \vdash A\rceil$ by sem. def.

□

From the fundamental lemma follows our main theorem normalization and subject reduction.

Theorem 7.14 (Normalization and Subject Reduction). *If* $\Gamma \vdash t : \tau$ *then* $t \searrow w$ *and* $\Gamma \vdash t \equiv w : \tau$

Proof. By the Fundamental theorem (Lemma 7.13), we have $\Gamma \Vdash t = t : \tau$ (choosing the identity substitution for θ and θ'). This includes a definition $t \searrow w$. Since w is in weak head normal form (i.e. whnf w), we have $w \searrow w$. Therefore, we can easily show that also $\Gamma \Vdash t = w : \tau$. By well-formedness (Lemma 6.8), we also have that $\Gamma \vdash t \equiv w : \tau$ and more specifically, $\Gamma \vdash w : \tau$. □

Using the fundamental lemma, we can also show that every term has a unique type. This requires first showing some standard inversion lemmas and then showing function type injectivity.

Lemma 7.15 (Inversion). *(1) If* $\Gamma \vdash x : \check{\tau}$ *then* $x : \check{\tau}' \in \Gamma$ *for some* $\check{\tau}'$ *and* $\Gamma \vdash \check{\tau} \equiv \check{\tau}' : u$.
(2) If $\Gamma \vdash fn\, y \Rightarrow t : \tau$ *then* $\Gamma, y : \check{\tau}_1 \vdash t : \tau_2$ *for some* $\check{\tau}_1, \tau_2$ *and If* $\Gamma \vdash \tau \equiv (y : \check{\tau}_1) \Rightarrow \tau_2 : u$.
(3) If $\Gamma \vdash t\ s : \tau$ *then* $\Gamma \vdash t : (y : \check{\tau}_1) \Rightarrow \tau_2$ *and* $\Gamma \vdash s : \check{\tau}_1$ *for some* $\check{\tau}_1$ *and* τ_2 *and* $\Gamma \vdash \tau \equiv \{s/y\}\tau_2 : u$.
(4) If $\Gamma \vdash \lceil C\rceil : \tau$ *then* $\Gamma \vdash \lceil C\rceil : \lceil T\rceil$ *for some contextual type* T *and* $\Gamma \vdash \tau \equiv \lceil T\rceil : u$.
(5) If $\Gamma \vdash rec^{\mathcal{I}}\ (b_v \mid b_{\mathtt{app}} \mid b_{\mathtt{lam}})\ \Psi\ t : \tau$ *where* $\mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (y : \lceil\Psi \vdash \mathsf{tm}\rceil) \Rightarrow \tau$ *then* $\Gamma \vdash t : \lceil\Psi \vdash \mathsf{tm}\rceil$ *and* $\Gamma \vdash \mathcal{I} : u$ *and* $\Gamma \vdash b_v : \mathcal{I}$ *and* $\Gamma \vdash b_{\mathtt{app}} : \mathcal{I}$ *and* $\Gamma \vdash b_{\mathtt{lam}} : \mathcal{I}$ *and* $\Gamma \vdash \tau \equiv \{\Psi/\psi, t/y\}\tau : u$.
(6) If $\Gamma \vdash u_1 : \tau$ *then there some* u_2 *s.t.* $(u_1, u_2) \in \mathfrak{A}$ *and* $\Gamma \vdash \tau \equiv u_2 : u$.
(7) If $\Gamma \vdash (y : \check{\tau}_1) \Rightarrow \tau_2 : \tau$ *then there is some* u_1, u_2, *and* u_3 *s.t.* $(u_1, u_2, u_3) \in \mathfrak{R}$ *and* $\Gamma \vdash \check{\tau}_1 : u_1, \Gamma, y : \check{\tau}_1 \vdash \tau_2 : u_2$ *and* $\Gamma \vdash u_3 \equiv \tau : u$.

Proof. By induction on the typing derivation. □

Lemma 7.16 (Injectivity of Function Type). *If* $\Gamma \vdash (y : \check{\tau}_1) \Rightarrow \tau_2 \equiv (y : \check{\tau}_1') \Rightarrow \tau_2' : u$ *then* $\Gamma \vdash \check{\tau}_1 \equiv \check{\tau}_1' : u_1$ *and* $\Gamma, y : \check{\tau}_1 \vdash \tau_2 \equiv \tau_2 : u_2$ *and* $(u_1, u_2, u) \in \mathfrak{R}$.

Proof. By the fundamental theorem (Lemma 7.13) $\Gamma \Vdash (y : \check{\tau}_1) \Rightarrow \tau_2 \equiv (y : \check{\tau}_1') \Rightarrow \tau_2' : u$ (choosing the identity substitution for θ and θ'). By the sem. equality def., we have $\Gamma \Vdash \check{\tau}_1 = \check{\tau}_1' : u_1$ and $\Gamma, y : \check{\tau}_1 \Vdash \tau_2 = \tau_2' : u_2$ and $(u_1, u_2, u) \in \mathfrak{R}$. By well-formedness of semantic typing (Lemma 6.8), we have $\Gamma \vdash \check{\tau}_1 \equiv \check{\tau}_1' : u_1$ and and $\Gamma, y : \check{\tau}_1 \vdash \tau_2 \equiv \tau_2' : u_2$ □

Theorem 7.17 (Type Uniqueness).

(1) If $\Gamma; \Psi \vdash M : A$ *and* $\Gamma; \Psi \vdash M : B$ *then* $\Gamma \vdash A \equiv B : \mathsf{type}$.
(2) If $\Gamma \vdash t : \check{\tau}$ *and* $\Gamma \vdash t : \check{\tau}'$ *then* $\Gamma \vdash \check{\tau} \equiv \check{\tau}' : u$.

Proof. By mutual induction on the typing derivation exploiting typing inversion lemmas. □

Last but not least, the fundamental lemma allows us to show that not every type is inhabited and thus Cocon can be used as a logic. To establish this stronger notion of consistency, we first prove that we can discriminate type constructors.

Lemma 7.18 (Type Constructor Discrimination). *Neutral types, sorts, and function types are can be discriminated.*

(1) If $\Gamma \vdash u_1 \equiv u_2 : u_3$ *then* $u_1 = u_2$ *(they are the same).*
(2) $\Gamma \vdash x\ \vec{t} \neq u : u'$*.*
(3) $\Gamma \vdash x\ \vec{t} \neq (y : \check{\tau}_1) \Rightarrow \tau_2$*.*
(4) $\Gamma \vdash x\ \vec{t} \neq \lceil T \rceil$*.*
(5) $\Gamma \vdash u \neq (y : \check{\tau}_1) \Rightarrow \tau_2$*.*

Proof. Proof by contradiction. To show for example that $\Gamma \vdash x\ \vec{t} \neq (y : \check{\tau}_1) \Rightarrow \tau_2$, we assume $\Gamma \vdash x\ \vec{t} \equiv (y : \check{\tau}_1) \Rightarrow \tau_2 : u$. By the fundamental lemma (Lemma 7.13), we have $\Gamma \Vdash x\ \vec{t} \equiv (y : \check{\tau}_1) \Rightarrow \tau_2 : u$ (choosing the identity substitution for θ and θ'); but this is impossible given the semantic equality definition (Fig. 20). ◻

Theorem 7.19 (Consistency). $x : u_0 \nvdash t : x$.

Proof. Assume $x : u_0 \vdash t : x$. By subject reduction (Lemma 7.14), there is some w s.t. $t \searrow w$ and $\Gamma \vdash t \equiv w : x$ and in particular, we must have $\Gamma \vdash w : x$. As x is neutral, it cannot be equal u, $(y : \check{\tau}_1) \Rightarrow \tau_2$, or $\lceil T \rceil$ (Lemma 7.18). Thus w can also not be a sort, function, or contextual object. Hence, w can only be neutral, i.e. given the assumption $x : u_0$, the term w must be x. This implies that $\Gamma \vdash x : x$ and implies $\Gamma \vdash x \equiv u_0 : u$ by inversion lemma for typing. But this is impossible by Lemma 7.18. ◻

8 RELATED WORK

HOAS within dependent type theory. We propose a new type theoretic foundation where LF is integrated within a Martin Löf type theory. This is in some sense a radical step. A more lightweight approach is to integrate at least some of the benefits of LF within an existing type theory. This is for example accomplished by weak HOAS approaches [Chlipala 2008; Despeyroux et al. 1995] where we get α-renaming for free but still have to deal with capture-avoiding substitutions. The Hybrid library [Felty and Momigliano 2012] in Coq goes further supporting both α-renaming and substitution by encoding a specification logic within Coq. However it is unclear whether these approaches scale to dependently typed encodings and can be integrated smoothly into practice.

Metaprogramming. Dependent type theory is flexible enough to serve as its own metaprogramming language. For this reason many dependently typed systems [Christiansen 2014, 2015; Ebner et al. 2017; van Der Walt and Swierstra 2012] try to support meta-programming in practice using quote operator to turn an expression into its syntactical representation and unquote operator to escape the quotation and refer to another computation whose value will be plugged in at its place. However, a clean theoretical foundation is missing.

Davies and Pfenning [2001] observed the similarity between modal types in S4 and quasi-quotation to support simply typed metaprogramming. However their work concentrated on closed simply-typed code. In Cocon, we can describe open pieces of code, i.e. code that depends on a context of assumptions, and work within a Martin Löf type theory. Hence, Cocon has the potential to provide a basis for dependently typed metaprogramming.

9 CONCLUSION

Cocon is a first step towards integrating LF methodology into Martin-Löf style dependent type theories and and bridges the longstanding gap between these two worlds. We have established type uniqueness, normalization, and consistency. The next immediate step is to derive an equivalence algorithm based on weak head reduction and show its completeness. We expect that this will follow a similar Kripke-style logical relation as the one we described. This would allow us to justify that type checking Cocon programs is decidable.

It should be possible to implement Cocon as an extension to Beluga– from a syntactic point of view, it would be a small change. It also seems possible to extend existing implementation of Agda, however this might be more work, as in this case one needs to implement the LF infrastructure.

ACKNOWLEDGMENTS

This material is based upon work supported by the Humboldt Foundation and NSERC (Natural Science and Engineering Research Council, Canada) No. mmmmmmm. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the funding agencies.

A APPENDIX

EXAMPLES

Cost-Semantics. As a first example, consider the definition of a cost semantics for our small term language described earlier. As we aim to reason by structural induction on the evaluation judgement, we define the evaluation judgment $m \Downarrow_k v$ which says that the term m evaluates in at most steps k to the value v as an inductive type:

```
inductive Eval : (m:⌈ ⊢ tm⌉)(n:⌈ ⊢ tm⌉)(k: nat) type =
| E_Lam : (m : ⌈x:tm ⊢ tm⌉)(k:nat)
          Eval ⌈ ⊢ lam λx.⌊m⌋⌉ ⌈ ⊢ lam λx.⌊m⌋⌉ k
| E_App : (m : ⌈ ⊢ tm⌉) (n: ⌈ ⊢ tm⌉)(k:nat)(l:nat)(j:nat)
          (m': ⌈x:tm ⊢ tm⌉)(v:⌈ ⊢ tm⌉)(w:⌈ ⊢ tm⌉)
          Eval m ⌈ ⊢ lam λx.⌊m'⌋⌉ k → Eval n v l → Eval ⌈ ⊢ ⌊ m'⌋ with ⌊v⌋/x ⌉ w j
        → Eval ⌈ ⊢ app ⌊m⌋ ⌊n⌋⌉ w (k + l + j + 1)
| E_Let : (m : ⌈ ⊢ tm⌉)(n: ⌈x:tm ⊢ tm⌉)(k:nat)(l:nat)(v : ⌈ ⊢ tm⌉) (w : ⌈ ⊢ tm⌉)
          Eval m v k
        → Eval ⌈ ⊢ ⌊n⌋ with ⌊v⌋/x ⌉ w l
        → Eval ⌈ ⊢ letv ⌊m⌋ λx.⌊n⌋ ⌉ w (k + l + 1)
```

We define here an inductive type that relates closed `tm`-objects `m` and `n` with the cost `k`. We rely on the inductive type `nat` in addition to functions such as addition on natural numbers. In a dependent type theory such as Coq or Agda, we would not be able to define an inductive type over (open) LF objects and exploit HOAS. In this example, we clearly state that we evaluate closed terms. To evaluate `⌈ ⊢ app ⌊m⌋ ⌊n⌋⌉` where `m` and `n` denote a closed `tm`-objects, we evaluate `m` and `n` respectively. Note that when we refer to variables inside a box (or quoted) expression, we need to first unbox (or unquote) them. We write the unboxing here as `⌊m⌋`.

In general, we are unboxing open terms, i.e. terms that may contain free variables. Hence, we are unboxing a term together with a substitution. For example, to evaluate the body `m'` where we replace the LF variable `x` with the closed term `v`, we unbox `m'` with the substitution `⌊v⌋/x`. This is written as `⌈ ⊢ ⌊m'⌋` **with** `⌊v⌋/x⌉`. In general, we may omit mentioning the substitution that is associated with every unbox-operation, if the substitution is simply the identity.

In Beluga, an inductive definition about open LF objects is possible, but we would not be able to compute `k + l + 1`, as Beluga is an indexed language – not a full dependently typed language. Therefore, we cannot refer to addition function in the index.

Compilation. As a simple example of compilation, we consider here a function `trans` which eliminates let-expressions. As we also must traverse the body of lambda-abstractions and let-expression, this function takes a term in the context γ as input and returns a term in the same context as output. As for example in Beluga, contexts are first-class in our language and we specify their shape using a context schema `tm_ctx` which states that the context only contains `tm` declarations.

```
rec trans: (γ : ⌈tm_ctx⌉) ⌈γ ⊢ tm⌉ → ⌈γ ⊢ tm⌉ =
fun  γ    (p : ⌈γ ⊢# tm⌉)        = p
   | γ   ⌈γ ⊢ app ⌊m⌋ ⌊n⌋⌉       = ⌈γ ⊢ app ⌊trans γ m⌋ ⌊trans γ n⌋⌉
   | γ   ⌈γ ⊢ lam λx.⌊m⌋⌉        = ⌈γ ⊢ lam λx. ⌊trans (γ,x:tm) m⌋ ⌉
   | γ   ⌈γ ⊢ letv ⌊m⌋ λy.⌊n⌋⌉   = ⌈γ ⊢ app (lam λx. ⌊trans (γ,x:tm) n⌋ ⌊trans γ m⌋ ⌉
```

We write the translation by pattern matching on the HOAS tree of type `⌈γ ⊢ tm⌉`. Four different cases arise. First, we might encounter a variable from γ. As in Beluga, we use a pattern variable `p` of type `⌈γ ⊢# tm⌉` which can only be instantiated with variables from γ. Second, we translate `⌈γ ⊢ app ⌊m⌋ ⌊n⌋⌉` by simply recursively translating `m` and `n` and rebuilding our term. Third, we translate `⌈γ ⊢ lam λx.⌊m⌋⌉` by translating `m`. Note that `m` has type `⌈γ, x:tm ⊢ tm⌉` and hence `trans m` returns a term in the context `γ, x:tm`. Last, we translate `⌈γ ⊢ letv ⌊m⌋ λy.⌊n⌋⌉` by translating each part and replacing the let-expression with the application of a lambda-abstraction.

The function `trans` is close to what we can already write in Beluga with one major differences: we are able to inline recursive calls such as `trans n` within a HOAS tree by supporting the boxing (quote) and the unboxing (unquote) of contextual objects; this is in contrast to Beluga where we are forced to write programs in a let-box style. Further, we only distinguish between LF variables that are bound by a λ-abstraction or in a LF context and computation-level variables. If computation-level variables have a contextual type, then we can use them to construct an LF object (or LF context) by unboxing them. In Beluga, we essentially distinguish between three different kinds of variables: LF variables, computation-level variables, and meta-variables (or contextual variables) that are of contextual type. Our treatment here unifies the latter two classes into one.

We now prove that the operational meaning of a term is preserved and it still evaluates in at most k steps. In other words, our optimization did not add any additional costs. This is done by recursively analyzing and pattern matching on the derivation `Eval m v k`. We write the type of each of the recursive calls as comments to illuminate what is happening in the background.

```
rec ctrns : ⌈ ⊢ tm⌉ → ⌈ ⊢ tm⌉ = fun m ⇒ trans ⌈ ⌉ m ;

rec val_preserve : (m:⌈ ⊢ tm⌉)(v:⌈ ⊢ tm⌉)(k : nat) Eval m v k → Eval (ctrns m) (ctrns v) k =
fun ⌈ ⊢ lam λx.⌊m⌋⌉ ⌈ ⊢ lam λx.⌊m⌋⌉ k (E_Lam m k) = E_Lam (trans ⌈x:tm⌉ m) k
  | ⌈ ⊢ app ⌊m⌋ ⌊n⌋ ⌉ w (k+l+j+1) (E_App m n k l j m' v w e1 e2 e3) =
    E_App (ctrns m) (ctrns n) k l j (trans ⌈x:tm⌉ m') (ctrns v) (ctrns w)
          (val_preserve m ⌈ ⊢ lam λx.⌊m'⌋⌉ k e1)
                      % Eval (ctrns m) ⌈ ⊢ lam λx.⌊trans ⌈x:tm⌉ m'⌋⌉ k
          (val_preserve n v l e2)                  % Eval (ctrns n) (ctrns v) l
          (subst ⌈ ⊢ tm⌉ (fun e ⇒ Eval e (ctrns w) j)
                (ctrns ⌈ ⊢ ⌊m'⌋ with ⌊v⌋/x⌉) (⌈ ⊢ ⌊trans ⌈x:tm⌉ m'⌋ with ⌊ctrns v⌋/x⌉)
                (lemma_trans ⌈⌉ m' v) (val_preserve ⌈ ⊢ ⌊m'⌋ with ⌊v⌋/x⌉ w j e3))
                      % Eval (ctrns ⌈ ⊢ ⌊m'⌋ with ⌊v⌋/x⌉) (ctrns w) j
  | ⌈ ⊢ letv ⌊m⌋ λy.⌊n⌋⌉ w (k+l+1) (E_Let m n k l v w e1 e2) =
    E_App ⌈ ⊢ lam λx. ⌊trans ⌈x:tm⌉ n⌋⌉ (ctrns m) 0 k l (trans ⌈x:tm⌉ n) (ctrns v) (ctrns w)
          (E_Lam (trans ⌈x:tm⌉ n) 0)
                                % Eval (ctrns ⌈ ⊢ lam λy.⌊n⌋⌉) (ctrns ⌈ ⊢ lam λy.⌊n⌋⌉) 0)
          (val_preserve m v k e1)                % Eval (ctrns m) (ctrns v) k)
          (subst ⌈ ⊢ tm⌉ (fun e ⇒ Eval e (ctrns w) l)
                (ctrns ⌈ ⊢ ⌊n⌋ with ⌊v⌋/x⌉) (⌈ ⊢ ⌊trans ⌈x:tm⌉ n⌋ with ⌊ctrns v⌋/x⌉)
                (lemma_trans ⌈⌉ n v) (val_preserve ⌈ ⊢ ⌊n⌋ with ⌊v⌋/x⌉ w l e2))
                                % Eval (ctrns ⌈ ⊢ ⌊n⌋ with ⌊v⌋/x⌉) (ctrns w) l)
```

The proof above relies on a lemma that states that it does not matter whether we translate first a term `m` of type `⌈γ, x:tm ⊢ tm⌉` and then replace `x` with the translation of the term `v` or we translate directly the term `m` where we already substituted for `x` the term `v`. It is applied by using substitutivity of identity type whose type is:
`subst : (A : type) (P : A → type) (a b : A) a = b → P a → P b`

```
rec lemma_trans : (γ : ctx) (m : ⌈γ, x:tm ⊢ tm⌉) (v : ⌈γ ⊢ tm⌉)
        trans γ ⌈γ ⊢ ⌊m⌋ with wkγ, ⌊v⌋/x⌉
        = ⌈γ ⊢ ⌊trans (γ, x:tm) m⌋ with wkγ, ⌊trans γ v⌋/x⌉ =
fun γ ⌈γ, x ⊢ x⌉ = refl % since trans γ v = ⌈γ ⊢ ⌊trans γ v ⌋⌉
  | γ ⌈γ, x ⊢ ⌊p⌋ with wkγ⌉ where p : ⌈γ ⊢# λtm⌉ =
      refl %trans γ p = ⌈γ ⊢ ⌊p⌋ with wkγ⌉
  | γ ⌈γ, x ⊢ lam λy.⌊m⌋⌉ =
    cong ⌈γ,y:tm ⊢ tm⌉ ⌈γ,y:tm ⊢ tm⌉
      (fun e ⇒ ⌈γ ⊢ lam (λy. ⌊e⌋)⌉)
      (trans (γ,y:tm) ⌈γ,y ⊢ ⌊m⌋ with wkγ,⌊v⌋/x,y/y⌉)
      ⌈γ,y ⊢ ⌊trans (γ,x:tm,y:tm) m⌋ with wkγ, ⌊trans γ v⌋/x, y/y⌉
      (lemma_trans (γ, y:tm) ⌈γ, y, x ⊢ ⌊m⌋ with wkγ,x/x, y/y⌉)

  | γ ⌈γ, x ⊢ app ⌊m⌋ ⌊n⌋⌉ =
    cong2 ⌈γ ⊢ tm⌉ ⌈γ ⊢ tm⌉ ⌈γ ⊢ tm⌉
      (fun m n ⇒ ⌈γ ⊢ app ⌊m⌋ ⌊n⌋⌉)
      (trans γ ⌈γ ⊢ ⌊m with wkγ,⌊v⌋/x⌋⌉)
      (⌈γ ⊢ ⌊trans (γ,x:tm) m with wkγ,⌊trans (γ) v⌋,x⌋⌉)
      (trans γ ⌈γ ⊢ ⌊n with wkγ,⌊v⌋/x⌋⌉)
      (⌈γ ⊢ ⌊trans (γ,x:tm) n with wkγ,⌊trans (γ) v⌋,x⌋⌉)
      (lemma_trans γ m v) (lemma_trans (γ) n v)

  | γ ⌈γ, x ⊢ letv ⌊m⌋ λy.⌊n⌋⌉ =
    cong2 ⌈γ ⊢ tm⌉ ⌈γ ⊢ tm⌉ ⌈γ ⊢ tm⌉
      (fun m n ⇒ ⌈γ ⊢ app (lam λy.⌊n⌋) ⌊m⌋⌉)
      (trans γ ⌈γ ⊢ ⌊m⌋ with wkγ,⌊v⌋/x⌉)
      ⌈γ ⊢ ⌊trans (γ,x:tm) m⌋ with wkγ,⌊trans γ v⌋/x⌉
```

```
(trans (γ,y:tm)⌈γ,y ⊢ ⌊n⌋ with wkγ,⌊v⌋/x, y/y⌉)
⌈γ,y ⊢ ⌊trans (γ,x:tm,y:tm) n⌋ with wkγ,⌊trans γ v⌋/x,y/y⌉
(lemma_trans γ m v)
(lemma_trans (γ,y:tm) ⌈γ, y, x ⊢ ⌊n⌋ with wkγ,x/x,y/y⌉)
```

In the first case, we have by definitional equality that indeed `trans γ v = ⌈γ ⊢ ⌊trans γ v ⌋⌉`. In the second case, we in addition exploit that composing $\mathtt{wk}_\gamma$, `⌊trans γ v⌋/x` with $\mathtt{wk}_\gamma$ simply is $\mathtt{wk}_\gamma$ effectively dropping `⌊trans γ v⌋/x`.

The recursive cases are handled by means of congruence. We rebuild the term on each side by joining together the equalities obtained for the subterms. The types for the functions on congruences are respectively:

```
cong1 : (A B : type) (f : A → B) (a1 a2 : A)→ a1 = a2 → (f a1) = (f a2)
cong2 : (A B C : type) (f : A → B → C) (a1 a2 : A) (b1 b2 : B) → a1 = a2 → b1 = b2 → (f a1 b1)=(f a2 b2)
```

In the `lam` case and in the `letv` case, we exploit that we can build a substitution that exchanges variables such that we can keep the `x`, the variable that we want to replace, to the right most position. This is a standard technique employed in HOAS systems.

Note that we would not be able to implement such functions in Beluga for several reasons:

- We directly refer to the function `ctrns` in the type of `val_preserve` to indicate that if `m` evaluates to a value `v` then the translation of `m` (i.e. `ctrns m`) also evaluates to the translation of the value of `v` (i.e. `ctrns v`). In Beluga, we would need to reify the function `ctrns` as an inductive type and then pass it as an additional argument to `val_preserve`.
- The lemma `lemma_trans` also directly references the function type of `trans` in stating that the equality property.
- We reason by equality, in particular we use the functions `subst`, `cong1`, and `cong2` which all are polymorphic. Polymorphism is presently not supported in Beluga, or any system we are aware of that supports HOAS.

RECURSORS OVER DEPENDENTLY TYPED LF OBJECTS

LF definitions are not inductive – however, we can generate recursors for each LF type following the procedure described in [Pientka and Abel 2015].

To illustrate concretely, how recursors look for a dependently typed LF signature, we consider here another example: the representation of well-typed terms. In this case, our LF signature contains the following type families and constants.

$$\begin{array}{lll} \text{LF Signature} & \Sigma ::= & \mathsf{tp} : \mathsf{type}, \mathsf{nat} : \mathsf{tp},\ \mathsf{arr} : \Pi y{:}\mathsf{tp}.\Pi y{:}\mathsf{tp}.\mathsf{tp} \\ & & \mathsf{tm} : \Pi a{:}\mathsf{tp}.\mathsf{type},\ \mathsf{z} : \mathsf{tm}\ \mathsf{nat},\ \mathsf{suc} : \Pi y{:}\mathsf{tm}\ \mathsf{nat}.\mathsf{tm}\ \mathsf{nat}, \\ & & \mathtt{lam} : \Pi a{:}\mathsf{tp}.\Pi b{:}\mathsf{tp}.\Pi y{:}(\Pi x{:}\mathsf{tm}\ a.\mathsf{tm}\ b).\mathsf{tm}\ (\mathsf{arr}\ a\ b), \\ & & \mathsf{app} : \Pi a{:}\mathsf{tp}.\Pi b{:}\mathsf{tp}.\Pi x{:}\mathsf{tm}\ (\mathsf{arr}\ a\ b).\Pi y{:}\mathsf{tm}\ a.\mathsf{tm}\ b \end{array}$$

For easier readability, we simply write how it looks when we declare these constants in Beluga or Twelf. Note that we write simply → if B does not depend on x in $\Pi x{:}A.B$. We also omit abstracting over the implicit arguments – this is common practice in logical frameworks, as the type for `A` and `B` can be inferred.

```
tp : type
nat: tp.
arr: tp → tp → tp.

tm: tp → type.
z  : tm nat.
suc: tm nat → tm nat.
lam: (tm A → tm B) → tm (arr A B).
app: tm (arr A B) → tm A → tm B.
```

To build the recursor for the type family tm a we proceed as follows:

- We generalize the recursor to $\mathrm{rec}^{\mathcal{I}}\ \mathcal{B}\ \Psi\ s\ t$. Here the intention is that s has type $\lceil\vdash \mathsf{tp}\rceil$ and t has type $\lceil\Psi \vdash \mathsf{tm}\ \lfloor s\rfloor_{\mathrm{wk}_{\hat\Psi}}\rceil$. Hence `t` depends not only on the LF context Ψ but also on the type `s`. As `s` denotes a closed type, we weaken it to be used within the LF context Ψ.
  In general, we have a vector $\vec{s}$ to describe all the implicit arguments t depends on. Note that even in an LF signature that is simply typed, i.e. we have for example defined tm : type, the type of t already depends on Ψ,

since it has contextual type $\lceil \Psi \vdash \mathsf{tm} \rceil$. Moreover, we already are tracking this dependency on the LF context. Hence, the generalization to more dependent arguments is quite natural.

- The recursor for iterating over contextual terms of type $\lceil \psi \vdash \mathsf{tm}\ a \rceil$ will have 5 branches: 4 branches covering each constructor and one branch for the variable case.

$$\mathsf{rec}^{\mathcal{I}}\ (b_v \mid b_{\mathsf{z}} \mid b_{\mathsf{suc}} \mid b_{\mathsf{app}} \mid b_{\mathsf{lam}})$$

  where

$$\begin{array}{lcl} b_v & ::= & \psi, a, p \Rightarrow t_v \\ b_{\mathsf{z}} & ::= & \psi \Rightarrow t_{\mathsf{z}} \\ b_{\mathsf{suc}} & ::= & \psi, n, f_n \Rightarrow t_{\mathsf{suc}} \\ b_{\mathsf{lam}} & ::= & \psi, a, b, m, f_m \Rightarrow t_{\mathsf{lam}} \\ b_{\mathsf{app}} & ::= & \psi, a, b, n, m, f_n, f_m \Rightarrow t_{\mathsf{app}} \end{array}$$

- We give the typing rules for the recursor over terms of type $\lceil \psi \vdash \mathsf{tm}\ a \rceil$. Each branch gives rise to a specific typing rule and we label the $\vdash_l$ with the label l where $l = \{v,\ \mathsf{z},\ \mathsf{suc},\ \mathsf{lam},\ \mathsf{app}\}$ for clarity.

Recursor over LF Terms $\mathcal{I} = (\psi : \mathsf{tm\_ctx}) \Rightarrow (a : \lceil \vdash \mathsf{tp} \rceil) \Rightarrow (y : \lceil \psi \vdash \mathsf{tm}\ a \rceil) \Rightarrow \tau$

where $l = \{v,\ \mathsf{z},\ \mathsf{suc},\ \mathsf{lam},\ \mathsf{app}\}$

$$\frac{\Gamma \vdash s : \lceil \vdash \mathsf{tp} \rceil \qquad \Gamma \vdash t : \lceil \Psi \vdash \mathsf{tm}\ \lfloor s \rfloor_{\mathsf{wk}_\Psi} \rceil \quad \Gamma \vdash \mathcal{I} : u \quad \Gamma \vdash_l b_l : \mathcal{I}}{\Gamma \vdash \mathsf{rec}^{\mathcal{I}}\ (b_v \mid b_{\mathsf{z}} \mid b_{\mathsf{suc}} \mid b_{\mathsf{lam}} \mid b_{\mathsf{app}})\ \Psi\ s\ t : \{\Psi/\psi,\ s/a,\ t/y\}\tau}$$

Branches where

$$\frac{\Gamma, \psi : \mathsf{tm\_ctx}, a : \lceil \vdash \mathsf{tp} \rceil,\ p : \lceil \psi \vdash_{\#} \mathsf{tm}\ \lfloor a \rfloor_{\mathsf{wk}_\psi} \rceil \vdash t_v : \{a/a,\ p/y\}\tau}{\Gamma \vdash_v (\psi, a, p \Rightarrow t_v) : \mathcal{I}}$$

$$\frac{\Gamma, \psi : \mathsf{tm\_ctx} \vdash t_{\mathsf{z}} : \{\lceil \vdash \mathsf{nat} \rceil/a,\ \lceil \psi \vdash \mathsf{z} \rceil/y\}\tau}{\Gamma \vdash_{\mathsf{z}} (\psi \Rightarrow t_{\mathsf{z}}) : \mathcal{I}}$$

$$\frac{\begin{array}{l} \Gamma\psi : \mathsf{tm\_ctx}, m : \lceil \psi \vdash \mathsf{tm\ nat} \rceil \\ \quad f_m : \{\lceil \vdash \mathsf{nat} \rceil/a,\ m/y\}\tau \qquad \vdash \quad t_{\mathsf{suc}} : \{\lceil \vdash \mathsf{nat} \rceil/a,\ \lceil \psi \vdash \mathsf{suc}\ \lfloor m \rfloor_{\mathsf{id}} \rceil/y\}\tau \end{array}}{\Gamma \vdash_{\mathsf{suc}} (\psi, m \Rightarrow t_{\mathsf{suc}}) : \mathcal{I}}$$

$$\frac{\begin{array}{l} \Gamma,\psi : \mathsf{tm\_ctx}, a : \lceil \vdash \mathsf{tp} \rceil,\ b : \lceil \vdash \mathsf{tp} \rceil, \\ \quad m : \lceil \psi \vdash \mathsf{tm}\ (\mathsf{arr}\ \lfloor a \rfloor_{\mathsf{wk}_\psi}\ \lfloor b \rfloor_{\mathsf{wk}_\psi}) \rceil, n : \lceil \psi \vdash \mathsf{tm}\ \lfloor a \rfloor_{\mathsf{id}} \rceil \\ \quad f_m : \{\mathsf{arr}\ \lfloor a \rfloor_{\mathsf{id}}\ \lfloor b \rfloor_{\mathsf{id}}/a, m/y\}\tau, f_n : \{a/a,\ n/y\}\tau \qquad \vdash \quad t_{\mathsf{app}} : \{b/a,\ \lceil \psi \vdash \mathsf{app}\ \lfloor m \rfloor_{\mathsf{id}}\ \lfloor n \rfloor_{\mathsf{id}} \rceil/y\}\tau \end{array}}{\Gamma \vdash_{\mathsf{app}} (\psi, a, b, m, n, f_n, f_m \Rightarrow t_{\mathsf{app}}) : \mathcal{I}}$$

$$\frac{\begin{array}{l} \Gamma,\phi : \mathsf{tm\_ctx}, a : \lceil \vdash \mathsf{tp} \rceil, b : \lceil \vdash \mathsf{tp} \rceil \\ \quad m : \lceil \phi, x : \mathsf{tm}\ \lfloor a \rfloor_{\mathsf{wk}_\psi} \vdash \mathsf{tm}\ \lfloor b \rfloor_{\mathsf{wk}_\psi} \rceil, \\ \quad f_m : \{(\phi, x : \mathsf{tm}\ a)/\psi, b/a,\ m/y\}\tau \qquad \vdash \quad t_{\mathtt{lam}} : \{\phi/\psi,\ \lceil \vdash \mathsf{arr}\ \lfloor a \rfloor_{\mathsf{id}}\ \lfloor b \rfloor_{\mathsf{id}} \rceil/a,\ \lceil \phi \vdash \mathtt{lam}\ \lambda x.\lfloor m \rfloor_{\mathsf{id}}\ \rceil/y\}\tau \end{array}}{\Gamma \vdash_{\mathtt{lam}} (\psi, a, b, m, f_m \Rightarrow t_{\mathtt{lam}}) : \mathcal{I}}$$