Journal of Automated Reasoning (2023) 67:36
https://doi.org/10.1007/s10817-023-09678-y

# Finitary Type Theories With and Without Contexts

**Philipp G. Haselwarter**[1,3] · **Andrej Bauer**[1,2]

Received: 1 November 2021 / Accepted: 12 July 2023 / Published online: 7 October 2023


**Abstract**
We give a definition of finitary type theories that subsumes many examples of dependent type theories, such as variants of Martin–Löf type theory, simple type theories, first-order and higher-order logics, and homotopy type theory. We prove several general meta-theorems about finitary type theories: weakening, admissibility of substitution and instantiation of metavariables, derivability of presuppositions, uniqueness of typing, and inversion principles. We then give a second formulation of finitary type theories in which there are no explicit contexts. Instead, free variables are explicitly annotated with their types. We provide translations between finitary type theories with and without contexts, thereby showing that they have the same expressive power. The context-free type theory is implemented in the nucleus of the Andromeda 2 proof assistant.

**Keywords** Dependent type theory · Context-free type theory · Formal meta-theory · Proof assistants

## 1 Introduction

We present a general definition of a class of dependent type theories which we call *finitary type theories*. In fact, we provide two variants of such type theories, with and without typing contexts, and show that they are equally expressive by providing translations between them. Our definition broadly follows the development of general type theories [6], but is specialized to serve as a formalism for implementation of a proof assistant. Indeed, the present paper is the theoretical foundation of the Andromeda 2 proof assistant, in which type theories are entirely defined by the user.

To be quite precise, we shall study *syntactic presentations* of type theories, in the sense that theories are seen as syntactic constructions, and the meta-theorems conquered by a frontal

✉ Philipp G. Haselwarter
philipp@haselwarter.org

Andrej Bauer
andrej@andrej.com

[1] Faculty of Mathematics and Physics, University of Ljubljana, Ljubljana, Slovenia

[2] Institute for Mathematics, Physics, and Mechanics, Ljubljana, Slovenia

[3] Aarhus University, Århus, Denmark

assault on abstract syntax. Even though this may not be the most fashionable approach to type theory, we were lead to it by our determination to understand precisely what we were implementing in Andromeda 2. We certainly expect that the syntactic presentations will match nicely with some of the modern semantic accounts of type theories, and that the usefulness of finitary type theories will transcend mere theoretical support for proof assistants.

We thus present our development of type theories in an elementary style, preferring concrete to abstract definitions and constructions, without compromising generality. In particular, this means that we first define "raw" terms, judgements, rules, and the like, and then proceed in stages to carve out the well-behaved fragment via predicates. Our motivations for this choice are fourfold. First, in practice type systems are defined in this fashion. Second, an elementary definition requires only very modest meta-mathematical foundations and lends itself to interpretation in various foundational systems. Third, by eschewing intermediate surrogates such as logical frameworks [23, 38] or quotient inductive-inductive types [2], the semantics of finitary type theories may be addressed directly, without recourse to the interpretation of such intermediates. And in any case, even the intermediates must eventually be syntactically presented if they are to be used at all. Fourth, the programming languages available to us are not sufficiently expressive to isolate the well-formed fragment of type theory in one fell swoop. They enable and insist on a more traditional approach, in which the input strings are converted to syntactic trees, and the type theoretic entities presented in their "raw" form, as values of inductively defined datatypes. The concrete nature of our constructions and meta-theorems then makes it possible to transcribe them to code in a straightforward fashion. Further discussion of alternative approaches is postponed to Sect. 7.

Our definition captures dependent type theories of Martin–Löf style, i.e. theories that strictly separate terms and types, have four judgement forms (for terms, types, type equations, and typed term equations), and hypothetical judgements standing in intuitionistic contexts. Among examples are the intensional and extensional Martin–Löf type theory, possibly with Tarski-style universes, homotopy type theory, Church's simple type theory, simply typed λ-calculi, and many others. A detailed presentation of first-order logic and Martin–Löf type theory as finitary type theories is available in [30, Appendix A], and [24, Appendix B] presents a finitary type theory for Harper's Equational LF [21], and encodes Gödel's System T in the logical framework. Counter-examples can be found just as easily: in cubical type theory the interval type is special, cohesive and linear type theories have non-intuitionistic contexts, polymorphic λ-calculi quantify over all types, pure type systems organize the judgement forms in their own way, and so on.

### Contributions

In Sect. 2 we give an account of dependent type theories that is close to how they are traditionally presented. A type theory should verify certain meta-theoretical properties: the constituent parts of any derivable judgement should be well-formed, substitution rules should be admissible, and each term should have a unique type. The definition of *finitary type theories* proceeds in stages. Each of the stages refines the notion of *rule* and *type theory* by specifying conditions of well-formedness. We start with the raw syntax (Sect. 2.1) of expressions and formal metavariables, out of which contexts, substitutions, and judgements are formed. Next we define *raw rules* (Sect. 2.3), a formal notion of what is commonly called "schematic inference rule". We introduce the *structural rules* (Figs. 4, 5, 6) that are shared by all type theories, and define *congruence rules* (Definition 2.17). These rules are then collected into raw type theories (Definition 2.21). The definition of raw rules ensures the well-typedness of each

constituent part of a raw rule, by requiring the derivability of the presuppositions of a rule. Next, we introduce *finitary rules* and *finitary type theories* (Sect. 2.4), whose rules form a well-founded order under which each rule is well-typed with respect to its predecessors. This way we rule out circularities in the derivations of well-typedness of rules, while the well-founded order provides an induction principle for finitary type theories. Finally, *standard* type theories are introduced (Definition 2.25) to enforce that each symbol is associated to a unique rule.

We prove the following meta-theorems about raw (Sect. 3.1), finitary (Sect. 3.2), and standard type theories (Sect. 3.3): admissibility of substitution and equality substitution (Theorem 3.8), admissibility of instantiation of metavariables (Theorem 3.13) and equality instantiation (Theorem 3.17), derivability of presuppositions (Theorem 3.18), admissibility of "economic" rules (Propositions 3.19, 3.20 and 3.22), inversion principles (Theorem 3.24), uniqueness of typing (Theorem 3.26).

The goal of Sect. 4 is the development of a context-free presentation of finitary type theories that can serve as foundation of the implementation of a proof assistant. The definition of finitary type theories in Sect. 2 is well-suited for the metatheoretic study of type theory, but does not directly lend itself to implementation. For instance, in keeping with traditional accounts of type theory, contexts are explicitly represented as lists.

In *context-free type theories*, the syntax of expressions (Sect. 4.1) is modified so that each free variable is annotated with its type $\mathsf{a}^A$ rather than being assigned a type by a context. As the variables occurring in the type annotation $A$ are also annotated, the dependency between variables is recorded. Judgements in context-free type theories thus do not carry an explicit context. Metavariables are treated analogously. To account for the possibility of proof-irrelevant rules like equality reflection, where not all of the variables used to derive the premises are recorded in the conclusion, we augment type and term equality judgements with *assumption sets* (Sect. 4.1.5). Intuitively, in a judgement $\vdash A \equiv B \ \mathsf{by}\ \alpha$, the assumption set $\alpha$ contains the (annotated) variables that were used in the derivation of the equation but may not be amongst the free variables of $A$ and $B$. The conversion rule of type theory allows the use of a judgemental equality to construct a term judgement. To ensure that assumption sets on equations are not lost as a result of conversion, we include *conversion terms* (Fig. 9).

Following the development of finitary type theories, we introduce raw context-free rules and type theories (Sect. 4.2). We proceed to define *context-free finitary rules* and type theories whose well-formedness is derivable with respect to a well-founded order (Definition 4.13), and *standard* theories (Definition 4.14).

Subsequently, we prove meta-theorems about context-free raw (Sect. 5.1), finitary (Sect. 5.2), and standard type theories (Sect. 5.3). The meta-theorems in this section are similar to those obtained for finitary type theories, with the exception of the meta-theorems specific to context-free type theories (Sect. 5.4). In particular, and contrary to finitary type theories, context-free raw type theories satisfy strengthening (Theorem 5.16). We further prove that conversion terms do not "get in the way" when working in context-free type theory (Theorem 5.17). The constructions underlying these meta-theorems are defined on judgements rather than derivations, and can thus be implemented effectively in a proof assistant for context-free type theories without storing derivation trees.

In Sect. 6, we establish a correspondence between type theories with and without contexts by constructing translations back and forth (Theorems 6.5 and 6.10).

## 2 Finitary Type Theories

Our treatment of type theories follows in essence the definition of general type theories carried out in [6], but is tailored to support algorithmic derivation checking in three respects: we limit ourselves to finitary symbols and rules, construe metavariables as a separate syntactic class rather than extensions of symbol signatures by fresh symbols, and take binding of variables to be a primitive operation on its own.

### 2.1 Raw Syntax

In this section we describe the *raw* syntax of fintary type theories, also known as pre-syntax. We operate at the level of *abstract binding trees*, i.e. we construe syntactic entities as syntax trees generated by grammatical rules in inductive fashion, and with all bound variables well-scoped. Of course, we still display such trees *concretely* as string of symbols, a custom that should not detract from the abstract view.

Raw expressions are formed without any typing discipline, but they have to be syntactically well-formed in the sense that free and bound variables must be well-scoped and that all symbols must be applied in accordance with the given signature. We shall explain the details of these conditions after a short word on notation.

We write $[X_1, \ldots, X_n]$ for a finite sequence and $f = \langle X_1 \mapsto Y_1, \ldots, X_n \mapsto Y_n \rangle$ for a sequence of pairs $(X_i, Y_i)$ that represents a map taking each $X_i$ to $Y_i$. An alternative notation is $\langle X_1 : Y_1, \ldots, X_n : Y_n \rangle$, and we may elide the parentheses $[\cdots]$ and $\langle \cdots \rangle$. The ***domain*** of such $f$ is the set $\mathsf{f} = \{X_1, \ldots, X_n\}$, and it is understood that all $X_i$ are different from one another. Given $X \notin \mathsf{f}$, the ***extension*** $\langle f, X \mapsto Y \rangle$ of $f$ by $X \mapsto Y$ is the map

$$\langle f, X \mapsto Y \rangle : Z \mapsto \begin{cases} Y & \text{if } Z = X, \\ f(Z) & \text{if } Z \in \mathsf{f}. \end{cases}$$

Given a list $\ell = [\ell_1, \ldots, \ell_n]$, we write $\ell_{(i)} = [\ell_1, \ldots, \ell_{i-1}]$ for its $i$-th initial segment. We use the same notation in other situations, for example $f_{(i)} = \langle X_1 \mapsto Y_1, \ldots, X_{i-1} \mapsto Y_{i-1} \rangle$ for $f$ as above.

#### 2.1.1 Variables and Substitution

We distinguish notationally between the disjoint sets of *free variables* $\mathsf{a}, \mathsf{b}, \mathsf{c}, \ldots$ and *bound variables* $x, y, z, \ldots$, each of which are presumed to be available in unlimited supply. The free variables are scoped by variable contexts, while the bound ones are always captured by abstractions.

The strict separation of free and bound variables is fashioned after *locally nameless syntax* [14, 28], a common implementation technique of variable binding in which free variables are represented as names and the bound ones as de Bruijn indices [17]. In Sect. 4 the separation between free and bound variables will be even more pronounced, as only the former ones are annotated with types.

We write $e[s/x]$ for the substitution of an expression $s$ for a bound variable $x$ in expression $e$ and $e[\vec{s}/\vec{x}]$ for the (parallel) substitution of $s_1, \ldots, s_n$ for $x_1, \ldots, x_n$, with the usual proviso about avoiding the capture of bound variables. In Sect. 3.1, when we prove admissibility of substitution, we shall also substitute expressions for free variables, which of course is written as $e[s/\mathsf{a}]$. Elsewhere we avoid such substitutions and only ever replace free variables by bound ones, in which case we write $e[x/\mathsf{a}]$. This typically happens when an expression

with a free variable is used as part of a binder, such as the codomain of a Π-type or the body of a lambda. We take care to always keep bound variables well-scoped under binders.

### 2.1.2 Arities and Signatures

The raw expressions of a finitary type theory are formed using ***symbols*** and ***metavariables***, which constitute two separate syntactic classes. Each symbol and metavariable has an associated arity, as follows.

The ***symbol arity*** $(c, [(c_1, n_1), \ldots, (c_k, n_k)])$ of a symbol $\mathsf{S}$ tells us that

1. the syntactic class of $\mathsf{S}$ is $c \in \{\mathsf{Ty}, \mathsf{Tm}\}$,
2. $\mathsf{S}$ accepts $k$ arguments,
3. the $i$-th argument must have syntactic class $c_i \in \{\mathsf{Ty}, \mathsf{Tm}, \mathsf{EqTy}, \mathsf{EqTm}\}$ and binds $n_i$ variables.

The syntactic classes Ty and Tm stand for type and term expressions, and EqTy and EqTm for type and term equations, respectively. For the time being the latter two are mere formalities, as the only expression of these syntactic classes are the dummy values $\star_{\mathsf{Ty}}$ and $\star_{\mathsf{Tm}}$. However, in Sect. 4 we will introduce genuine expressions of syntactic classes EqTy and EqTm.

***Example 2.1*** The arity of a type constant such as bool is $(\mathsf{Ty}, [])$, the arity of a binary term operation such as $+$ is $(\mathsf{Tm}, [(\mathsf{Tm}, 0), (\mathsf{Tm}, 0)])$. The arity of a quantifier such as the dependent product Π is $(\mathsf{Ty}, [(\mathsf{Ty}, 0), (\mathsf{Ty}, 1)])$ because it is a type former taking two type arguments, with the second one binding one variable, and the arity of a dependent function λ is $(\mathsf{Tm}, [(\mathsf{Ty}, 0), (\mathsf{Ty}, 1), (\mathsf{Tm}, 1)])$.

The ***metavariable arity*** associated to a metavariable $\mathsf{M}$ is a pair $(c, n)$, where the syntactic class $c \in \{\mathsf{Ty}, \mathsf{Tm}, \mathsf{EqTy}, \mathsf{EqTm}\}$ indicates whether $\mathsf{M}$ is respectively a type, term, type equality, or term equality metavariable, and $n$ is the number of term arguments it accepts. The metavariables of syntactic classes Ty and Tm are the ***object*** metavariables, and can be used to form expressions. The metavariables of syntactic classes EqTy and EqTm are the ***equality*** metavariables, and do not participate in formation of expressions. We introduce them to streamline several definitions, and to have a way of referring to equational premises in Sect. 4. The information about metavariable arities is collected in a metavariable context, cf. Sect. 2.1.4.

A metavariable $\mathsf{M}$ of arity $(c, n)$ could be construed as a symbol of arity

$$(c, [\underbrace{(\mathsf{Tm}, 0), \ldots, (\mathsf{Tm}, 0)}_{n}]).$$

This approach is taken in [6], but we keep metavariables and symbols separate because they play different roles, especially in context-free type theories in Sect. 4.

The information about symbol and metavariable arities is respectively collected in a ***symbol signature*** and a ***metavariable signature***, which map symbols and metavariables to their arities. When discussing syntax, it is understood that such signature have been given, even if we do not mention them explicitly. In particular, whenever expressions are formed in a given metavariable context, as described below, it is assumed that the metavariable signature is the one induced by the context.

| | | | |
|---|---|---|---|
| Type expression $A, B$ | $::=$ | $\mathsf{S}(e_1, \ldots, e_n)$ | type symbol application |
| | $\mid$ | $\mathsf{M}(t_1, \ldots, t_n)$ | type metavariable application |
| Term expression $s, t$ | $::=$ | $\mathsf{a}$ | free variable |
| | $\mid$ | $x$ | bound variable |
| | $\mid$ | $\mathsf{S}(e_1, \ldots, e_n)$ | term symbol application |
| | $\mid$ | $\mathsf{M}(t_1, \ldots, t_n)$ | term metavariable application |
| Argument $e$ | $::=$ | $A$ | type argument |
| | $\mid$ | $t$ | term argument |
| | $\mid$ | $\star_{\mathsf{Ty}}$ | dummy argument |
| | $\mid$ | $\star_{\mathsf{Tm}}$ | dummy argument |
| | $\mid$ | $\{x\}e$ | abstracted arg. ($x$ bound in $e$) |
| Judgement thesis $j$ | $::=$ | $A$ type | $A$ is a type |
| | $\mid$ | $t : A$ | $t$ has type $T$ |
| | $\mid$ | $A \equiv B$ by $\star_{\mathsf{Ty}}$ | $A$ and $B$ are equal types |
| | $\mid$ | $s \equiv t : A$ by $\star_{\mathsf{Tm}}$ | $s$ and $t$ are equal terms at $A$ |
| Abstracted judgement $\mathcal{J}$ | $::=$ | $j$ | judgement thesis |
| | $\mid$ | $\{x{:}A\}\ \mathcal{J}$ | abstracted jdgt. ($x$ bound in $\mathcal{J}$) |
| Boundary thesis $\mathcal{b}$ | $::=$ | $\square$ type | a type |
| | $\mid$ | $\square : A$ | a term of type $A$ |
| | $\mid$ | $A \equiv B$ by $\square$ | type equation boundary |
| | $\mid$ | $s \equiv t : B$ by $\square$ | term equation boundary |
| Abstracted boundary $\mathcal{B}$ | $::=$ | $\mathcal{b}$ | boundary thesis |
| | $\mid$ | $\{x{:}A\}\ \mathcal{B}$ | abstracted bdry. ($x$ bound in $\mathcal{B}$) |
| Variable context $\Gamma$ | $::=$ | $[\mathsf{a}_1{:}A_1, \ldots, \mathsf{a}_n{:}A_n]$ | |
| Metavariable context $\Theta$ | $::=$ | $[\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$ | |
| Hypothetical judgement | | $\Theta;\ \Gamma \vdash \mathcal{J}$ | |
| Hypothetical boundary | | $\Theta;\ \Gamma \vdash \mathcal{B}$ | |

**Fig. 1** The raw syntax of expressions, boundaries and judgements

### 2.1.3 Raw Expressions

The raw syntactic constituents of a finitary type theory, with respect to given symbol and metavariable signatures, are outlined in Fig. 1. In this section we discuss the top part of the figure, which involves the syntax of term and type expressions, and arguments.

A ***type expression***, or just a ***type***, is formed by an application $\mathsf{S}(e_1, \ldots, e_n)$ of a type symbol to arguments, or an application $\mathsf{M}(t_1, \ldots, t_n)$ of a type metavariable to term expressions. A ***term expression***, or just a ***term***, is a free variable $\mathsf{a}$, a bound variable $x$, an application $\mathsf{S}(e_1, \ldots, e_n)$ of a term symbol to arguments, or an application $\mathsf{M}(t_1, \ldots, t_n)$ of a term metavariable to term expressions.

An ***argument*** is a type or a term expression, the dummy argument $\star_{\mathsf{Ty}}$ of syntactic class $\mathsf{EqTy}$, or the dummy argument $\star_{\mathsf{Tm}}$ of syntactic class $\mathsf{EqTm}$. We write just $\star$ when it is clear which of the two should be used. Another kind of argument is an ***abstraction*** $\{x\}e$, which binds $x$ in $e$. An iterated abstraction $\{x_1\}\{x_2\}\cdots\{x_n\}e$ is abbreviated as $\{\vec{x}\}e$. Note

that abstraction is a primitive syntactic operation, and that it provides no typing information about $x$.

***Example 2.2*** In our notation a dependent product is written as $\Pi(A, \{x\}B)$, and a fully annotated function as $\lambda(A, \{x\}B, \{x\}e)$. The fact that $x$ ranges over $A$ is not part of the raw syntax and will be specified later by an inference rule.

In all cases, in order for an expression to be well-formed, the arities of symbols and metavariables must be respected. If $\mathsf{S}$ has arity $(c, [(c_1, n_1), \ldots, (c_k, n_k)])$, then it must be applied to $k$ arguments $e_1, \ldots, e_k$, where each $e_i$ is of the form $\{x_1\} \cdots \{x_{n_i}\} e_i'$ with $e_i'$ a non-abstracted argument of syntactic class $c_i$. Similarly, a metavariable $\mathsf{M}$ of arity $(c, n)$ must be applied to $n$ term expressions. When a symbol $\mathsf{S}$ takes no arguments, we write the corresponding expression as $\mathsf{S}$ rather than $\mathsf{S}()$, and similarly for metavariables.

As is usual, expressions which differ only in the choice of names of bound variables are considered syntactically equal, e.g., $\{x\}\mathsf{S}(\mathsf{a}, x)$ and $\{y\}\mathsf{S}(\mathsf{a}, y)$ are syntactically equal and we may write $(\{x\}\mathsf{S}(\mathsf{a}, x)) = (\{y\}\mathsf{S}(\mathsf{a}, y))$.

For future reference we define in Fig. 2 the sets of free variable, bound variable, and metavariable occurrences, where we write set comprehension as $\{\!|\cdots|\!\}$ in order to distinguish it from abstraction. A syntactic entity is said to be ***closed*** if no free variables occur in it.

### 2.1.4 Judgements and Boundaries

The bottom part of Fig. 1 displays the syntax of judgements and boundaries, which we discuss next.

There are four ***judgement forms***: "$A$ $\mathsf{type}$" asserts that $A$ is a type; "$t : A$" that $t$ is a term of type $A$; "$A \equiv B$ $\mathsf{by}$ $\star_{\mathsf{Ty}}$" that types $A$ and $B$ are equal; and "$s \equiv t : A$ $\mathsf{by}$ $\star_{\mathsf{Tm}}$" that terms $s$ and $t$ of type $A$ are equal. We may shorten the equational forms to "$A \equiv B$" and "$s \equiv t : A$" in this section, as the only possible choice for $\mathsf{by}$ is $\star$.

Less familiar, but equally fundamental, is the notion of a ***boundary***. Whereas a judgement is an assertion, a boundary is a *question* to be answered, a *promise* to be fulfilled, or a *goal* to be accomplished: "$\square$ $\mathsf{type}$" asks that a type be constructed; "$\square : A$" that the type $A$ be inhabited; and "$A \equiv B$ $\mathsf{by}$ $\square$" and "$s \equiv t : A$ $\mathsf{by}$ $\square$" that equations be proved.

An ***abstracted judgement*** has the form $\{x{:}A\}\ \mathcal{J}$, where $A$ is a type expression and $\mathcal{J}$ is a (possibly abstracted) judgement. The variable $x$ is bound in $\mathcal{J}$ but not in $A$. Thus in general an abstracted judgement has the form

$$\{x_1{:}A_1\} \cdots \{x_n{:}A_n\}\ j,$$

where $j$ is a ***judgement thesis***, i.e. an expression taking one of the four (non-abstracted) judgement forms. We may abbreviate such an abstraction as $\{\vec{x}{:}\vec{A}\}\ j$. Analogously, an ***abstracted boundary*** has the form

$$\{x_1{:}A_1\} \cdots \{x_n{:}A_n\}\ \mathcal{b},$$

where $\mathcal{b}$ is a ***boundary thesis***, i.e. it takes one of the four (non-abstracted) boundary forms. The reason for introducing abstracted judgements and boundaries will be explained shortly.

An abstracted boundary has the associated metavariable arity

$$\mathsf{ar}(\{x_1{:}A_1\} \cdots \{x_n{:}A_n\}\ \mathcal{b}) = (c, n)$$

**Free variables:**

$$\mathsf{fv}(\mathsf{a}) = \{\!|\mathsf{a}|\!\} \qquad \mathsf{fv}(x) = \{\!|\}\!\} \qquad \mathsf{fv}(\{x\}e) = \mathsf{fv}(e)$$
$$\mathsf{fv}(\mathsf{S}(e_1 \dots e_n)) = \mathsf{fv}(e_1) \cup \dots \cup \mathsf{fv}(e_n)$$
$$\mathsf{fv}(\mathsf{M}(t_1 \dots t_n)) = \mathsf{fv}(t_1) \cup \dots \cup \mathsf{fv}(t_n)$$

$$\mathsf{fv}(A\ \mathsf{type}) = \mathsf{fv}(A) \qquad \mathsf{fv}(t : A) = \mathsf{fv}(t) \cup \mathsf{fv}(A)$$
$$\mathsf{fv}(A \equiv B\ \mathsf{by}\ \star) = \mathsf{fv}(A) \cup \mathsf{fv}(B)$$
$$\mathsf{fv}(s \equiv t : A\ \mathsf{by}\ \star) = \mathsf{fv}(s) \cup \mathsf{fv}(t) \cup \mathsf{fv}(A)$$
$$\mathsf{fv}(\{x : A\}\ \mathcal{J}) = \mathsf{fv}(A) \cup \mathsf{fv}(\mathcal{J})$$

$$\mathsf{fv}(\square\ \mathsf{type}) = \{\!|\}\!\} \qquad \mathsf{fv}(\square : A) = \mathsf{fv}(A)$$
$$\mathsf{fv}(A \equiv B\ \mathsf{by}\ \square) = \mathsf{fv}(A) \cup \mathsf{fv}(B)$$
$$\mathsf{fv}(s \equiv t : A\ \mathsf{by}\ \square) = \mathsf{fv}(s) \cup \mathsf{fv}(t) \cup \mathsf{fv}(A)$$
$$\mathsf{fv}(\{x : A\ \}\mathcal{B}) = \mathsf{fv}(A) \cup \mathsf{fv}(\mathcal{B})$$

**Exposed bound variables:**

$$\mathsf{bv}(\mathsf{a}) = \{\!|\}\!\} \qquad \mathsf{bv}(x) = \{\!|x|\!\} \qquad \mathsf{bv}(\{x\}e) = \mathsf{bv}(e) \setminus \{\!|x|\!\}$$
$$\mathsf{bv}(\mathsf{S}(e_1 \dots e_n)) = \mathsf{bv}(e_1) \cup \dots \cup \mathsf{bv}(e_n)$$
$$\mathsf{bv}(\mathsf{M}(t_1 \dots t_n)) = \mathsf{bv}(t_1) \cup \dots \cup \mathsf{bv}(t_n)$$

**Metavariables:**

$$\mathsf{mv}(\mathsf{a}) = \{\!|\}\!\} \qquad \mathsf{mv}(x) = \{\!|\}\!\} \qquad \mathsf{mv}(\{x\}e) = \mathsf{mv}(e)$$
$$\mathsf{mv}(\mathsf{S}(e_1 \dots e_n)) = \mathsf{mv}(e_1) \cup \dots \cup \mathsf{mv}(e_n)$$
$$\mathsf{mv}(\mathsf{M}(t_1 \dots t_n)) = \{\!|\mathsf{M}|\!\} \cup \mathsf{mv}(t_1) \cup \dots \cup \mathsf{mv}(t_n)$$

$$\mathsf{mv}(A\ \mathsf{type}) = \mathsf{mv}(A) \qquad \mathsf{mv}(t : A) = \mathsf{mv}(t) \cup \mathsf{mv}(A)$$
$$\mathsf{mv}(A \equiv B\ \mathsf{by}\ \star) = \mathsf{mv}(A) \cup \mathsf{mv}(B)$$
$$\mathsf{mv}(s \equiv t : A\ \mathsf{by}\ \star) = \mathsf{mv}(s) \cup \mathsf{mv}(t) \cup \mathsf{mv}(A)$$
$$\mathsf{mv}(\{x{:}A\}\ \mathcal{J}) = \mathsf{mv}(A) \cup \mathsf{mv}(\mathcal{J})$$

$$\mathsf{mv}(\square\ \mathsf{type}) = \{\!|\}\!\} \qquad \mathsf{mv}(\square : A) = \mathsf{mv}(A)$$
$$\mathsf{mv}(A \equiv B\ \mathsf{by}\ \square) = \mathsf{mv}(A) \cup \mathsf{mv}(B)$$
$$\mathsf{mv}(s \equiv t : A\ \mathsf{by}\ \square) = \mathsf{mv}(s) \cup \mathsf{mv}(t) \cup \mathsf{mv}(A)$$
$$\mathsf{mv}(\{x : A\}\ \mathcal{B}) = \mathsf{mv}(A) \cup \mathsf{mv}(\mathcal{B})$$

**Fig. 2** Free, bound, and metavariable occurrences

where $c \in \{\mathsf{Ty}, \mathsf{Tm}, \mathsf{EqTy}, \mathsf{EqTm}\}$ is the syntactic class of $\mathcal{b}$. Similarly, the associated metavariable arity of an argument is

$$\mathsf{ar}(\{x_1\} \cdots \{x_n\}e) = (c, n)$$

where $c \in \{\mathsf{Ty}, \mathsf{Tm}\}$ is the syntactic class of the (non-abstracted) expression $e$.

**Filling the placeholder with a head:**

$$(\square\ \mathsf{type})\boxed{A} = (A\ \mathsf{type})$$
$$(\square : A)\boxed{t} = (t : A)$$
$$(A \equiv B\ \mathsf{by}\ \square)\boxed{e} = (A \equiv B\ \mathsf{by}\ \star)$$
$$(s \equiv t : A\ \mathsf{by}\ \square)\boxed{e} = (s \equiv t : A\ \mathsf{by}\ \star)$$
$$(\{x{:}A\}\ \mathcal{B})\boxed{\{x\}e} = (\{x{:}A\}\ \mathcal{B}\boxed{e}).$$

**Filling the placeholder with an equality:**

$$(\square\ \mathsf{type})\boxed{A_1 \equiv A_2\ \mathsf{by}\ \star} = (A_1 \equiv A_2\ \mathsf{by}\ \star),$$
$$(\square : A)\boxed{t_1 \equiv t_2\ \mathsf{by}\ \star} = (t_1 \equiv t_2 : A\ \mathsf{by}\ \star),$$
$$(\{x{:}A\}\ \mathcal{B})\boxed{e_1 \equiv e_2\ \mathsf{by}\ \star} = (\{x{:}A\}\ \mathcal{B}\boxed{e_1 \equiv e_2\ \mathsf{by}\ \star}),$$

**Fig. 3** Filling the placeholder of a boundary

The placeholder $\square$ in a boundary $\mathcal{B}$ may be ***filled*** with an argument $e$, called the ***head***, to give a judgement $\mathcal{B}\boxed{e}$, provided that the arities of $\mathcal{B}$ and $e$ match. Because equations are proof irrelevant, their placeholders can be filled uniquely with (suitably abstracted) dummy value $\star$. Filling is summarized in Fig. 3, where we also include notation for filling an object boundary with an equation that results in the corresponding equation. The figure rigorously explicates the dummy values, but we usually omit them. Filling may be inverted: given an abstracted judgement $\mathcal{J}$ there is a unique abstracted boundary $\mathcal{B}$ and a unique argument $e$ such that $\mathcal{J} = \mathcal{B}\boxed{e}$.

***Example 2.3*** If the symbols $\mathsf{A}$ and $\mathsf{Id}$ have arities

$$(\mathsf{Ty}, []), \quad \text{and} \quad (\mathsf{Ty}, [(\mathsf{Ty}, 0), (\mathsf{Tm}, 0), (\mathsf{Tm}, 0)]),$$

respectively, then the boundaries

$$\{x{:}\mathsf{A}\}\{y{:}\mathsf{A}\}\ \square : \mathsf{Id}(\mathsf{A}, x, y) \qquad \text{and} \qquad \{x{:}\mathsf{A}\}\{y{:}\mathsf{A}\}\ x \equiv y : \mathsf{A}\ \mathsf{by}\ \square$$

may be filled with heads $\{x\}\{y\}x$ and $\{x\}\{y\}\star$ to yield abstracted judgements

$$\{x{:}\mathsf{A}\}\{y{:}\mathsf{A}\}\ x : \mathsf{Id}(\mathsf{A}, x, y) \qquad \text{and} \qquad \{x{:}\mathsf{A}\}\{y{:}\mathsf{A}\}\ x \equiv y : \mathsf{A}\ \mathsf{by}\ \star.$$

Names of bound variables are immaterial, we would still get the same judgement if we filled the left-hand boundary with $\{u\}\{v\}u$ or $\{y\}\{x\}y$, but not with $\{x\}\{y\}y$.

Information about available metavariables is collected by a ***metavariable context***, which is a finite list $\Theta = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$, also construed as a map, assigning to each metavariable $\mathsf{M}_i$ a boundary $\mathcal{B}_i$. In Sect. 2.3, the assigned boundaries will assign the typing of metavariable, while at the level of raw syntax they determine metavariable arities. That is, $\Theta$ assigns the metavariable arity $\mathsf{ar}(\mathcal{B}_i)$ to $\mathsf{M}_i$.

A metavariable context $\Theta = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$ may be *restricted* to a metavariable context $\Theta_{(i)} = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_{i-1}{:}\mathcal{B}_{i-1}]$.

The metavariable context $\Theta$ is syntactically well formed when each $\mathcal{B}_i$ is a syntactically well-formed boundary over $\Sigma$ and the metavariable signature induced by $\Theta_{(i)}$. In addition each $\mathcal{B}_i$ must be closed, i.e. contain no free variables.

A ***variable context*** $\Gamma = [\mathsf{a}_1{:}A_1, \ldots, \mathsf{a}_n{:}A_n]$ over a metavariable context $\Theta$ is a finite list of pairs written as $\mathsf{a}_i{:}A_i$. It is considered syntactically valid when the variables $\mathsf{a}_1, \ldots, \mathsf{a}_n$ are all distinct, and for each $i$ the type expression $A_i$ is valid with respect to the signature and the metavariable arities assigned by $\Theta$, and the free variables occurring in $A_i$ are among $\mathsf{a}_1, \ldots, \mathsf{a}_{i-1}$. A variable context $\Gamma$ yields a finite map, also denoted $\Gamma$, defined by $\Gamma(\mathsf{a}_i) = A_i$.

A ***context*** is a pair $\Theta; \Gamma$ consisting of a metavariable context $\Theta$ and a variable context $\Gamma$ over $\Theta$. A syntactic entity is considered syntactically valid over a signature and a context $\Theta; \Gamma$ when all symbol and metavariable applications respect the assigned arities, the free variables are among $|\Gamma|$, and all bound variables are properly abstracted. It goes without saying that we always require all syntactic entities to be valid in this sense.

A ***(hypothetical) judgement*** has the form

$$\Theta; \Gamma \vdash \mathcal{J}.$$

It differs from traditional notion of a judgement in a non-essential way, which nevertheless requires an explanation. First, the context of a hypothetical judgement

$$\Theta; \mathsf{a}_1{:}A_1, \ldots, \mathsf{a}_m{:}A_m \vdash \{x_1{:}B_1\} \cdots \{x_m{:}B_m\}\ j$$

provides information about metavariables, not just the free variables. Second, the variables are split between the context $\mathsf{a}_1{:}A_1, \ldots, \mathsf{a}_n{:}A_n$ on the left of $\vdash$, and the abstraction $\{x_1{:}B_1\} \cdots \{x_m{:}B_m\}$ on the right. It is useful to think of the former as the *global* hypotheses that interact with other judgements, and the latter as *local* to the judgement. We could of course delegate the metavariable context to be part of the signature as is done in [6], and revert to the more familiar form

$$\mathsf{a}_1{:}A_1, \ldots, \mathsf{a}_n{:}A_n, x_1{:}B_1, \ldots, x_m{:}B_m \vdash j$$

by joining the variable context and the abstraction, but we would still have to carry the metavariable information in the signature, and would lose the ability to explicitly mark the split between the global and the local parts. The split will be especially important in Sect. 4, where the context will be removed, but the abstraction kept.

***Hypothetical boundaries*** are formed in the same fashion, as

$$\Theta; \Gamma \vdash \mathcal{B}.$$

The intended meaning is that $\mathcal{B}$ is a well-typed boundary in context $\Theta; \Gamma$.

### 2.1.5 Metavariable Instantiations

Metavariables are slots that can be instantiated with arguments. Suppose $\Theta = \langle \mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_k{:}\mathcal{B}_k \rangle$ is a metavariable context over a symbol signature $\Sigma$. An ***instantiation of*** $\Theta$ ***over*** a context $\Xi; \Gamma$ is a seqence $I = \langle \mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_k \mapsto e_k \rangle$, representing a map that takes each $\mathsf{M}_i$ to an argument $e_i$ over $\Xi; \Gamma$ such that $\mathsf{ar}(\mathcal{B}_i) = \mathsf{ar}(e_i)$.

An instantiation $I = \langle \mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_k \mapsto e_k \rangle$ of $\Theta$ may be *restricted* to an instantiation $I_{(i)} = \langle \mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_{i-1} \mapsto e_{i-1} \rangle$ of $\Theta_{(i)}$.

An instantiation $I$ of $\Theta$ over $\Xi; \Gamma$ acts on a term- or type-expression $u$ over $\Theta; \Delta$ to give an expression $I_* u$ in which the metavariables are replaced by expressions, as follows:

$$I_* x = x, \qquad I_* \mathsf{a} = \mathsf{a}, \qquad I_* \star = \star, \qquad I_*(\{x\}e) = \{x\}(I_* e),$$
$$I_*(\mathsf{S}(e_1, \ldots, e_n)) = \mathsf{S}(I_* e_1, \ldots, I_* e_n),$$
$$I_*(\mathsf{M}_i(t_1, \ldots, t_{n_i})) = e_i[(I_* t_1)/x_1, \ldots, (I_* t_{n_i})/x_{n_i}].$$

Here, the symbol $\mathsf{S}$ and metavariable $\mathsf{M}_i$ take $n$ and $n_i$ arguments respectively. The instantiated expression $I_* u$ is valid for $\Xi; \Gamma, I_* \Delta$. Abstracted judgements and boundaries may be

instantiated too:

$$I_*(A\ \mathsf{type}) = (I_*A\ \mathsf{type}), \qquad I_*(t : A) = (I_*t : I_*A),$$
$$I_*(A \equiv B\ \mathsf{by}\ \star) = (I_*A \equiv I_*B\ \mathsf{by}\ \star), \qquad I_*(\{x{:}A\}\ \mathcal{J}) = \{x{:}I_*A\}\ I_*\mathcal{J},$$

and by imagining that $I_*\Box = \Box$, the reader can tell how to instantiate a boundary. Finally, a hypothetical judgement $\Theta; \Delta \vdash \mathcal{J}$ may be instantiated to $\Xi; \Gamma, I_*\Delta \vdash I_*\mathcal{J}$, and similarly for a hypothetical boundary.

### 2.2 Deductive Systems

We briefly recall the notions of a deductive system, derivability, and a derivation tree; see for example [1, 37] for background material. A ***(finitary) closure rule*** on a set $S$ is a pair $([p_1, \ldots, p_n], q)$, also displayed as

$$\frac{p_1 \ \cdots \ p_n}{q},$$

where $\{p_1, \ldots, p_n\} \subseteq S$ are the ***premises*** and $q \in S$ is the ***conclusion***. Let $\mathsf{Clos}(S)$ be the set of all closure rules on $S$.

A ***deductive system*** (also called a *closure system*) on a set $S$ is a family of closure rules $C : R \to \mathsf{Clos}(S)$, indexed by a set $R$ of rule names. A set $D \subseteq S$ is said to be ***deductively closed for*** $C$ when, for all $i \in R$, if $C_i = ([p_1, \ldots, p_n], q)$ and $\{p_1, \ldots, p_n\} \subseteq D$, then $q \in D$. The ***associated closure operator*** is the map $\mathcal{P}S \to \mathcal{P}S$ which takes $D \subseteq S$ to the least deductively closed supserset $\overline{D}$ of $D$, which exists by Tarski's fixed-point theorem [36]. We say that $q \in S$ is ***derivable from hypotheses*** $H \subseteq S$ when $q \in \overline{H}$, and that it is ***derivable in*** $C$ when $q \in \overline{\emptyset}$.

A closure rule $([p_1, \ldots, p_n], q)$ is ***admissible for*** $C$ when $\{p_1, \ldots, p_k\} \subseteq \overline{\emptyset}$ implies $q \in \overline{\emptyset}$. Note that adjoining an admissible closure rule to a closure system may change its associated closure operator. In contrast, nothing changes if we adjoin a ***derivable closure rule***, which is a rule $([p_1, \ldots, p_n], q)$ such that $q \in \overline{\{p_1, \ldots, p_n\}}$.

Derivability is witnessed by well-founded trees, which are constructed as follows. For each $q \in S$ let $\mathsf{Der}_C(q)$ be generated inductively by the clause:

- for every $i \in R$, if $C_i = ([p_1, \ldots, p_n], q)$ and $t_j \in \mathsf{Der}_C(p_j)$ for all $j = 1, \ldots, n$, then $\mathsf{der}_i(t_1, \ldots, t_n) \in \mathsf{Der}_C(q)$, where $\mathsf{der}$ is a formal tag (a "constructor").

The elements of $\mathsf{Der}_C(q)$ are ***derivation trees*** with ***conclusion*** $q$. Indeed, we may view $\mathsf{der}_i(t_1, \ldots, t_n)$ as a tree with the root labeled by $i$ and the subtrees $t_1, \ldots, t_n$. A leaf is a tree of the form $\mathsf{der}_j()$, which arises when the corresponding closure rule $C_j$ has no premises.

**Proposition 2.4** *Given a closure system $C$ on $S$, an element $q \in S$ is derivable in $C$ if, and only if, there exists a derivation tree over $C$ whose conclusion is $q$.*

***Proof*** The claim is that $T = \{q \in S \mid \exists t \in \mathsf{Der}_C(q)\,.\,\top\}$ coincides with $\overline{C}$. The inclusion $\overline{C} \subseteq T$ holds because $T$ is deductively closed. The reverse inclusion $T \subseteq \overline{C}$ is established by induction on derivation trees. □

We remark that allowing infinitary closure rules brings with it the need for the axiom of choice, for it is unclear how to prove that $T$ is deductively closed without the aid of choice.

It is evident that derivability and derivation trees are monotone in all arguments: if $S \subseteq S'$, $R \subseteq R'$, and the closure system $C' : R' \to \mathsf{Clos}(S')$ restricts to $C : R \to \mathsf{Clos}(S)$, then any

$q \in S$ derivable in $C$ is also derivable in $C'$ as an element of $S'$. Moreover, any derivation tree in $\mathsf{Der}_C(q)$ may be construed as a derivation tree in $\mathsf{Der}_{C'}(q)$.

Henceforth we shall consider solely deductive systems on the set of hypothetical judgements and boundaries. Because we shall vary the deductive system, it is useful to write $\Theta; \Gamma \vdash_C \mathcal{J}$ when $(\Theta; \Gamma \vdash \mathcal{J}) \in \overline{C}$, and similarly for $\Theta; \Gamma \vdash_C \mathcal{B}$.

### 2.3 Raw Rules and Type Theories

A type theory in its basic form is a collection of closure rules. Some closure rules are specified directly, but many are presented by *inference rules*—templates whose instantiations yield the closure rules. We deal with the *raw* syntactic structure of such rules first.

**Definition 2.5** A ***raw rule*** over a symbol signature $\Sigma$ is a hypothetical judgement over $\Sigma$ of the form $\Theta; [\,] \vdash j$. We notate such a raw rule as

$$\Theta \Longrightarrow j.$$

The elements of $\Theta$ are the ***premises*** and $j$ is the ***conclusion***. We say that the rule is an ***object rule*** when $j$ is a type or a term judgement, and an ***equality rule*** when $j$ is an equality judgement.

Defining inference rules as hypothetical judgements with empty contexts and empty abstractions permits in many situations uniform treatment of rules and judgements. Note that the premises and the conclusion may not contain any free variables, and that the conclusion must be non-abstracted. Neither condition impedes expressivity of raw rules, because free variables and abstractions may be promoted to premises.

***Example 2.6*** To help the readers' intuition, let us see how Definition 2.5 captures a traditional inference rule, such as product formation

$$\textsc{Ty-}\Pi\ \frac{\vdash \mathsf{A}\ \mathsf{type} \qquad \vdash \{x{:}\mathsf{A}\}\ \mathsf{B}(x)\ \mathsf{type}}{\vdash \Pi(\mathsf{A}, \{x\}\mathsf{B}(x))\ \mathsf{type}}$$

The use of $\mathsf{A}$ and $\mathsf{B}$ in the premises reveals that their arities are $(\mathsf{Ty}, 0)$, and $(\mathsf{Ty}, 1)$, respectively. In fact, the premises assign boundaries to metavariables: each premise is a boundary filled with a particular head, namely a generically applied metavariable. If we pull out the metavariables from the heads of premises, the assignment becomes explicit:

$$\frac{\mathsf{A}{:}\ (\Box\ \mathsf{type}) \qquad \mathsf{B}{:}\ (\{x{:}\mathsf{A}\}\ \Box\ \mathsf{type})}{\Pi(\mathsf{A}, \{x\}\mathsf{B}(x))\ \mathsf{type}}$$

This is just a different way of writing the raw rule

$$\mathsf{A}{:}(\Box\ \mathsf{type}),\ \mathsf{B}{:}(\{x{:}\mathsf{A}\}\ \Box\ \mathsf{type}) \Longrightarrow \Pi(\mathsf{A}, \{x\}\mathsf{B}(x))\ \mathsf{type}.$$

***Example 2.7*** We may translate raw rules back to their traditional form by filling the heads with metavariables applied to the variables they abstracts over. For example, the reader may readily verify that the raw rule

$$\begin{gathered}\mathsf{A}{:}(\Box\ \mathsf{type}),\ \mathsf{B}{:}(\{x{:}A\}\ \Box\ \mathsf{type}),\ \mathsf{b}{:}(\{x{:}A\}\ \Box : \mathsf{B}(x)) \\ \Longrightarrow \\ \lambda(\mathsf{A}, \{x\}\mathsf{B}(x), \{x\}\mathsf{b}(x)) : \Pi(\mathsf{A}, \{x\}\mathsf{B}(x))\end{gathered}$$

corresponds to the lambda introduction rule of dependent type theory that is traditionally written as

$$\textsc{Tm- }\lambda\quad \frac{\vdash \mathsf{A}\ \mathsf{type} \qquad \vdash \{x{:}\mathsf{A}\}\ \mathsf{B}(x)\ \mathsf{type} \qquad \vdash \{x{:}\mathsf{A}\}\ \mathsf{b}(x) : \mathsf{B}(x)}{\vdash \lambda(\mathsf{A}, \{x\}\mathsf{B}(x), \{x\}\mathsf{b}(x)) : \Pi(\mathsf{A}, \{x\}\mathsf{B}(x))}$$

Metavariables occurring as arguments to symbols, such as $\{x\}\mathsf{B}(x)$ in the conclusion of the previous example, are often abstracted and immediately applied. We record this pattern in the following definition.

**Definition 2.8** The ***generic application*** $\widehat{\mathsf{M}}$ of the metavariable $\mathsf{M}$ with associated boundary $\mathcal{B}$ is defined as:

1. $\widehat{M} = \{x_1\}\cdots\{x_k\}\,\mathsf{M}(x_1, \ldots, x_k)$ if $\mathsf{ar}(\mathcal{B}) = (c, k)$ and $c \in \{\mathsf{Ty}, \mathsf{Tm}\}$,
2. $\widehat{M} = \{x_1\}\cdots\{x_k\}\star$ if $\mathsf{ar}(\mathcal{B}) = (c, k)$ and $c \in \{\mathsf{EqTy}, \mathsf{EqTm}\}$.

Using generic metavariable applications, we can write the conclusion of Tm-λ more concisely as $\vdash \lambda(\widehat{\mathsf{A}}, \widehat{\mathsf{B}}, \widehat{\mathsf{b}}) : \Pi(\widehat{\mathsf{A}}, \widehat{\mathsf{B}})$, where we note that $\widehat{\mathsf{A}} = \mathsf{A}$.

***Example 2.9*** An informal presentation of type theory might specify the result type of applying $\mathsf{f}$ to $\mathsf{a}$ as "$\mathsf{B}$ with $\mathsf{a}$ substituted for $x$", i.e. $\mathsf{B}[\mathsf{a}/x]$. Since substitution is not part of the syntax of raw type theories but defined as a meta-operation, such a formulation would be nonsensical in our setting. The raw rule for application with full typing annotations on $\mathsf{app}$ can be written as follows.

$$\textsc{Tm- app}\quad \frac{\vdash \mathsf{A}\ \mathsf{type} \qquad \vdash \{x{:}A\}\ \mathsf{B}(x)\ \mathsf{type} \qquad \vdash \mathsf{f} : \Pi(\mathsf{A}, \widehat{\mathsf{B}}) \qquad \vdash \mathsf{a} : \mathsf{A}}{\vdash \mathsf{app}(\mathsf{A}, \widehat{\mathsf{B}}, \mathsf{f}, \mathsf{a}) : \mathsf{B}(\mathsf{a})}$$

Instead of using substitution, we define the type of the application as the metavariable application $\mathsf{B}(\mathsf{a})$, which is syntactically well-formed since $\mathsf{ar}(\mathsf{B}) = (\mathsf{Ty}, 1)$ in the above rule.

***Example 2.10*** Raw rules can also describe how to derive equality judgements. For instance, the raw rule

$$\mathsf{A}{:}(\square\ \mathsf{type}),\ \mathsf{s}{:}(\square : \mathsf{A}),\ \mathsf{t}{:}(\square : \mathsf{A}),\ \mathsf{p}{:}(\square : \mathsf{Id}(\mathsf{A}, \mathsf{s}, \mathsf{t})) \Longrightarrow \mathsf{s} \equiv \mathsf{t} : \mathsf{A}\ \mathsf{by}\ \star$$

corresponds to the *equality reflection* rule of extensional type theory that is traditionally written as

$$\textsc{Eq- Reflect}\quad \frac{\vdash \mathsf{A}\ \mathsf{type} \qquad \vdash \mathsf{s} : \mathsf{A} \qquad \vdash \mathsf{t} : \mathsf{A} \qquad \vdash \mathsf{p} : \mathsf{Id}(\mathsf{A}, \mathsf{s}, \mathsf{t})}{\vdash \mathsf{s} \equiv \mathsf{t} : \mathsf{A}}$$

For everyone's benefit, we shall display raw rules in traditional form, but use Definition 2.5 when formalities demand so.

***Example 2.11*** A rule that combines several aspects of the previous examples is $\beta$-reduction.

$$\textsc{EqTm- }\beta\quad \frac{\vdash \mathsf{A}\ \mathsf{type} \qquad \vdash \{x{:}A\}\ \mathsf{B}(x)\ \mathsf{type} \qquad \vdash \{x{:}\mathsf{A}\}\ \mathsf{b}(x) : \mathsf{B}(x) \qquad \vdash \mathsf{a} : \mathsf{A}}{\vdash \mathsf{app}(\mathsf{A}, \widehat{\mathsf{B}}, \lambda(\mathsf{A}, \widehat{\mathsf{B}}, \widehat{\mathsf{b}}), \mathsf{a}) \equiv \mathsf{b}(\mathsf{a}) : \mathsf{B}(\mathsf{a})}$$

Just like in Tm- app, we use metavariable application $\mathsf{b}(\mathsf{a})$ to describe the result of the $\beta$-reduction. Once the raw rule is instantiated into a closure rule, this application will be "activated" into a substitution.

It may be mystifying that there is no variable context $\Gamma$ in a raw rule, for is it not the case that rules may be applied in arbitrary contexts? Indeed, *closure* rules have contexts, but raw rules do not because they are just templates. The context appears once we instantiate the template, as follows.

**Definition 2.12** An ***instantiation*** of a raw rule $R = (\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n \implies \mathcal{b}\boxed{e})$ over context $\Theta; \Gamma$ is an instantiation $I = \langle \mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_n \mapsto e_n \rangle$ of its premises over $\Theta; \Gamma$. The associated ***closure rule*** $I_* R$ is $([p_1, \ldots, p_n, q], r)$ where $p_i$ is $\Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{e_i}$, $q$ is $\Theta; \Gamma \vdash I_*\mathcal{b}$, and $r$ is $\Theta; \Gamma \vdash I_*(\mathcal{b}\boxed{e})$.

We included among the premises the well-formedness of the instantiated boundary $\Theta; \Gamma \vdash I_*\mathcal{b}$, so that the conclusion is well-formed. We need the premise as an induction hypothesis in the proof of Theorem 3.18. In Sect. 3.2 we shall formulate well-formedness conditions that allow us to drop the boundary premise.

Of special interest are the rules that give type-theoretic meaning to primitive symbols. To define them, we need the boundary analogue of raw rules.

**Definition 2.13** A ***raw rule-boundary*** over a symbol signature $\Sigma$ is a hypothetical boundary over $\Sigma$ of the form $\Theta; [\,] \vdash \mathcal{b}$. We notate such a raw rule-boundary as

$$\Theta \implies \mathcal{b}.$$

The elements of $\Theta$ are the ***premises*** and $\mathcal{b}$ is the ***conclusion boundary***. We say that the rule-boundary is an ***object rule-boundary*** when $\mathcal{b}$ is a type or a term boundary, and an ***equality rule-boundary*** when $\mathcal{b}$ is an equality boundary.

Here is how a rule-boundary generates a rule associated to a symbol.

**Definition 2.14** Given a raw object rule-boundary

$$\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n \implies \mathcal{b}$$

over $\Sigma$, the ***associated symbol arity*** is $(c, [\mathsf{ar}(\mathcal{B}_1), \ldots, \mathsf{ar}(\mathcal{B}_n)])$, where $c \in \{\mathsf{Ty}, \mathsf{Tm}\}$ is the syntactic class of $\mathcal{b}$. The ***associated symbol rule*** for $\mathsf{S} \notin |\Sigma|$ is the raw rule

$$\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n \implies \mathcal{b}\boxed{\mathsf{S}(\widehat{\mathsf{M}}_1, \ldots, \widehat{\mathsf{M}}_n)}$$

over the extended signature $\langle \Sigma, \mathsf{S} \mapsto (c, [\mathsf{ar}(\mathcal{B}_1), \ldots, \mathsf{ar}(\mathcal{B}_n)]) \rangle$, where $\widehat{\mathsf{M}}_i$ is the generic application of the metavariable $\mathsf{M}_i$ with associated boundary $\mathcal{B}_i$. A raw rule is said to be a ***symbol rule*** if it is the associated symbol rule for some symbol $\mathsf{S}$.

The above definition is motivated by the observation that the head of the conclusion of a symbol rule has a particular shape, which can be calculated from its rule-boundary. The definition thus only requires the specification of the necessary data. Instead of describing how to construct a symbol rule given a rule boundary and symbol, we could have defined them directly as raw rules with conclusion heads of a particular form, but that would be less economical, since we would have to write out the conclusion in full, and we would still have to verify that the supplied head is the expected one. In examples we shall continue to display symbol rules in their traditional form.

***Example 2.15*** According to Definition 2.14, the symbol rule for $\Pi$ is generated by the rule-boundary

$$\frac{\vdash \mathsf{A}\ \mathsf{type} \qquad \vdash \{x{:}\mathsf{A}\}\ \mathsf{B}(x)\ \mathsf{type}}{\vdash \square\ \mathsf{type}}$$

Indeed, the associated symbol rule for $\Pi$ is

$$\frac{\vdash \mathsf{A}\ \mathsf{type} \qquad \vdash \{x{:}\mathsf{A}\}\ \mathsf{B}(x)\ \mathsf{type}}{\vdash \Pi(\mathsf{A}, \{x\}\mathsf{B}(x))\ \mathsf{type}}$$

We allow equational premises in object rules. For example,

$$\frac{\textsc{Refl'} \quad \vdash \mathsf{A}\ \mathsf{type} \qquad \vdash \mathsf{s} : \mathsf{A} \qquad \vdash \mathsf{t} : \mathsf{A} \qquad \vdash \mathsf{s} \equiv \mathsf{t} : \mathsf{A}}{\vdash \mathsf{refl}(\mathsf{A}, \mathsf{s}, \mathsf{t}, \star) : \mathsf{Id}(\mathsf{A}, \mathsf{s}, \mathsf{t})}$$

is a valid symbol rule, assuming $\mathsf{Id}$ and $\mathsf{refl}$ have their usual arities.

We also record the analogous construction of an equality rule from a given equality rule-boundary.

**Definition 2.16** Given an equality rule-boundary

$$\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n \Longrightarrow \mathcal{b},$$

the ***associated equality rule*** is

$$\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n \Longrightarrow \mathcal{b}\boxed{\star}.$$

We next formulate the rules that all type theories share, starting with the most nitty-gritty ones, the congruence rules.

**Definition 2.17** The ***congruence rules*** associated with a raw object rule $R$

$$\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n \Longrightarrow \mathcal{b}\boxed{e}$$

are closure rules, for any

$$I = \langle \mathsf{M}_1 \mapsto f_1, \ldots, \mathsf{M}_n \mapsto f_n \rangle \quad \text{and} \quad J = \langle \mathsf{M}_1 \mapsto g_1, \ldots, \mathsf{M}_n \mapsto g_n \rangle,$$

of the form

$$\frac{\begin{array}{ll} \Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i} & \text{for } i = 1, \ldots, n \\ \Theta; \Gamma \vdash (J_{(i)*}\mathcal{B}_i)\boxed{g_i} & \text{for } i = 1, \ldots, n \\ \Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i \equiv g_i} & \text{for object boundary } \mathcal{B}_i \\ \Theta; \Gamma \vdash I_*B \equiv J_*B & \text{if } \mathcal{b} = (\Box : B) \end{array}}{\Theta; \Gamma \vdash (I_*\mathcal{b})\boxed{I_*e \equiv J_*e}}$$

In case of a term equation at type $B$, the congruence rule has the additional premise $\Theta; \Gamma \vdash I_*B \equiv J_*B$, which ensures that the right-hand side of the conclusion $J_*e$ has type $I_*B$. Having the equation available as a premise allows us to use it in the inductive proof of Theorem 3.18. In Sect. 3.2 we show that the rule without the premises is derivable under suitable conditions.

***Example 2.18*** The congruence rule associated with the product formation rule from Example 2.6 is

$$\frac{\begin{array}{ll} \Theta; \Gamma \vdash A_1\ \mathsf{type} & \Theta; \Gamma \vdash \{x{:}A_1\}\ B_1\ \mathsf{type} \\ \Theta; \Gamma \vdash A_2\ \mathsf{type} & \Theta; \Gamma \vdash \{x{:}A_2\}\ B_2\ \mathsf{type} \\ \Theta; \Gamma \vdash A_1 \equiv A_2 & \Theta; \Gamma \vdash \{x{:}A_1\}\ B_1 \equiv B_2 \end{array}}{\Theta; \Gamma \vdash \Pi(A_1, \{x\}B_1) \equiv \Pi(A_2, \{x\}B_2)} \tag{2.1}$$

Next we have formation and congruence rules for the metavariables. As metavariables are like symbols whose arguments are terms, it is not suprising that their rules are quite similar to symbol rules.

**Definition 2.19** Given a context $\Theta; \Gamma$ over $\Sigma$ with $\Theta = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$, and $\mathcal{B}_k = (\{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \mathcal{b})$, the ***metavariable rules*** for $\mathsf{M}_k$ are the closure rules of the form

$$\text{TT- Meta}\quad \frac{\begin{array}{l}\Theta(\mathsf{M}_k) = \{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \mathcal{b} \\ \Theta; \Gamma \vdash t_j : A_j[\vec{t}_{(j)}/\vec{x}_{(j)}] \quad \text{for } j = 1, \ldots, m \\ \Theta; \Gamma \vdash \mathcal{b}[\vec{t}/\vec{x}]\end{array}}{\Theta; \Gamma \vdash (\mathcal{b}[\vec{t}/\vec{x}])\boxed{\mathsf{M}_k(\vec{t})}}$$

where $\vec{x} = (x_1, \ldots, x_m)$ and $\vec{t} = (t_1, \ldots, t_m)$. Recall that $\vec{t}_{(j)}$ stands for $[t_1, \ldots, t_{j-1}]$. In the second line of premises, we thus substitute the preceding term arguments $t_1, \ldots, t_{j-1}$ for the bound variables $x_1, \ldots, x_{j-1}$ in each type $A_j$. The last premise ensures the well-formedness of the boundary of the conclusion, just like the definition of the closure rule associated to a raw rule (Def. 2.12).

Furthermore, if $\mathcal{b}$ is an object boundary, then the ***metavariable congruence rules*** for $\mathsf{M}_k$ are the closure rules of the form

$$\text{TT- Meta- Congr}\quad \frac{\begin{array}{ll}\Theta(\mathsf{M}_k) = \{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \mathcal{b} & \\ \Theta; \Gamma \vdash s_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \Theta; \Gamma \vdash t_j : A_j[\vec{t}_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \Theta; \Gamma \vdash s_j \equiv t_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \Theta; \Gamma \vdash C[\vec{s}/\vec{x}] \equiv C[\vec{t}/\vec{x}] & \text{if } \mathcal{b} = (\square : C)\end{array}}{\Theta; \Gamma \vdash (\mathcal{b}[\vec{s}/\vec{x}])\boxed{\mathsf{M}_k(\vec{s}) \equiv \mathsf{M}_k(\vec{t})}}$$

where $\vec{s} = (s_1, \ldots, s_m)$ and $\vec{t} = (t_1, \ldots, t_m)$.

***Example 2.20*** If we collect the metavariables $\mathsf{A}$ and $\mathsf{B}$ introduced by the premises of the product formation rule from Example 2.6 into a metavariable context $\Theta = [\mathsf{A} : \square\ \mathsf{type}, \mathsf{B} : \{x{:}\mathsf{A}\}\ \square\ \mathsf{type}]$, we can apply the metavariable rule TT- Meta to derive that $\mathsf{B}(\mathsf{a})$ is a well-formed type under the context $\Theta; \mathsf{a} : \mathsf{A}$.

$$\frac{\Theta(\mathsf{B}) = \{x{:}A\}\ \square\ \mathsf{type} \qquad \Theta; \mathsf{a} : \mathsf{A} \vdash \mathsf{a} : \mathsf{A} \qquad \Theta; \mathsf{a} : \mathsf{A} \vdash \square\ \mathsf{type}}{\Theta; \mathsf{a} : \mathsf{A} \vdash \mathsf{B}(\mathsf{a})\ \mathsf{type}}$$

We are finally ready to give a definition of type theory which is sufficient for explaining derivability.

**Definition 2.21** A ***raw type theory*** $T$ over a signature $\Sigma$ is a family of raw rules over $\Sigma$, called the ***specific rules*** of $T$. The ***associated deductive system*** of $T$ consists of:

1. the ***structural rules*** over $\Sigma$:
   (a) the *variable*, *metavariable*, *metavariable congruence*, and *abstraction* closure rules (Fig. 4),
   (b) the *equality* closure rules, (Fig. 5),
   (c) the *boundary* closure rules (Fig. 6);

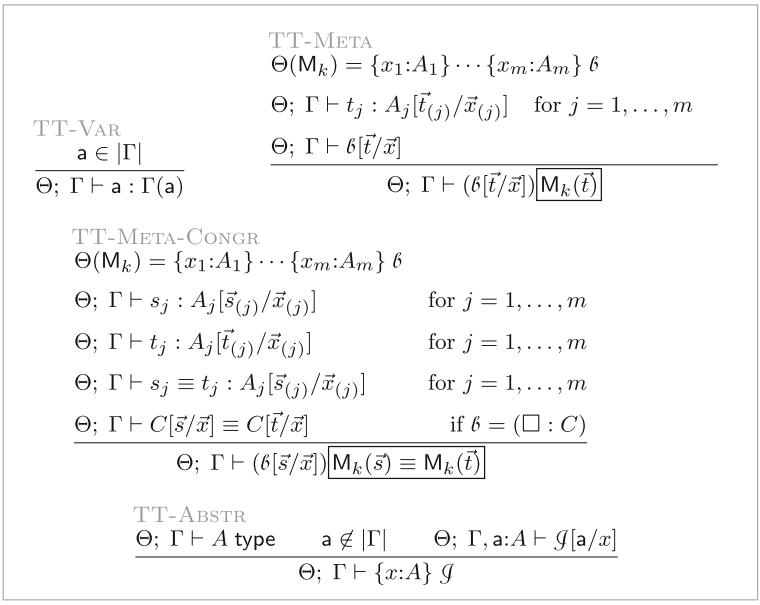


**Fig. 4** Variable, metavariable and abstraction closure rules


**Fig. 5** Equality closure rules

2. the instantiations of the specific rules of $T$ (Definition 2.12);
3. for each specific object rule of $T$, the instantiations of the associated congruence rule (Definition 2.17).

We write $\Gamma \vdash_T \mathcal{J}$ when $\Gamma \vdash \mathcal{J}$ is derivable with respect to the deductive system associated to $T$, and similarly for $\Gamma \vdash_T \mathcal{B}$.

Several remarks are in order regarding the above definition and the rules in Figs. 4, 5 and 6:

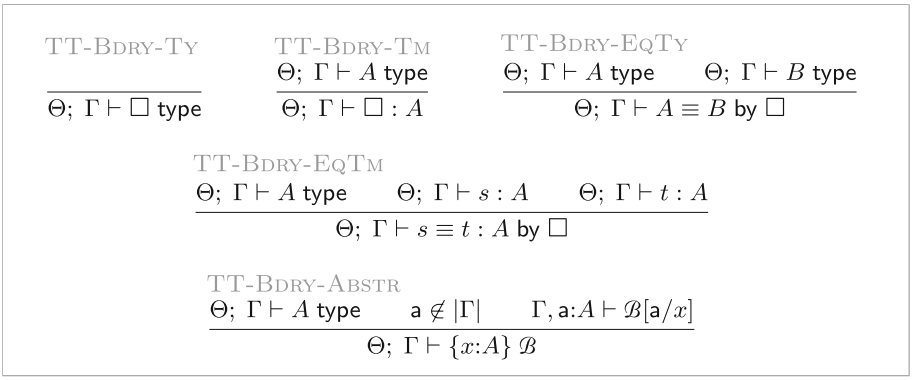


**Fig. 6** Well-formed abstracted boundaries

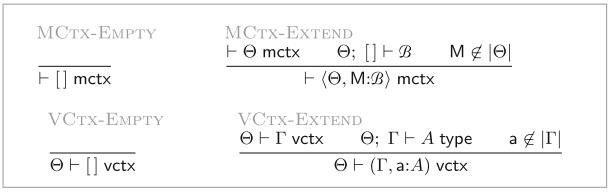


**Fig. 7** Well-formed metavariable and variable contexts

1. It is assumed throughout that all the entities involved are syntactically valid, i.e. that arities are respected and variables are well-scoped.
2. The metavariable rules TT- Meta and TT- Meta- Congr are exactly as in Definition 2.19.
3. The rules TT- Var, TT- Meta, and TT- Abstr contain *side-conditions*, such as $\mathsf{a} \in |\Gamma|$ and $\Theta(\mathsf{M}) = \{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \mathcal{B}$. For purely aesthetic reasons, these are written where premises ought to stand. For example, the correct way to read TT- Abstr is: "For all $\Theta$, $\Gamma$, $A$, $\mathsf{a}$, $\mathcal{J}$, if $\mathsf{a} \notin |\Gamma|$, then there is a closure rule with premises $\Theta;\ \Gamma \vdash A\ \mathsf{type}$ and $\Theta;\ \Gamma, \mathsf{a}{:}A \vdash \mathcal{J}[\mathsf{a}/x]$, and the conclusion $\Theta;\ \Gamma \vdash \{x{:}A\}\ \mathcal{J}$."
4. The structural rules impose no well-typedness conditions on contexts. Instead, Fig. 7 provides two auxiliary judgement forms, "$\vdash \Theta\ \mathsf{mctx}$" and "$\Theta \vdash \Gamma\ \mathsf{vctx}$", stating that $\Theta$ is a well-typed metavariable context, and $\Gamma$ a well-typed variable context over $\Theta$, respectively. These will be used as necessary. Note that imposing the additional premise $\Theta;\ \Gamma \vdash \Gamma(\mathsf{a})\ \mathsf{type}$ in TT- Var (where $\Gamma(\mathsf{a})$ is the type assigned to $\mathsf{a}$ by $\Gamma$) would *not* ensure well-formednes of $\Gamma$, as not all variables need be accessed in a derivation. Requiring that TT- Meta check the boundary of the metavariable is similarly ineffective.
5. We shall show in Sect. 3.1 that substitution rules (Fig. 8) are admissible.

This may be a good moment to record the difference between derivability and admissibility.

**Definition 2.22** Consider a raw theory $T$ and a raw rule $R$, both over a symbol signature $\Sigma$:

1. $R$ is ***derivable*** in $T$ when $R$ qua judgement has a derivation in $T$.
2. $R$ is ***admissible*** in $T$ when, for every instantiation $I$ of $R$, if the premises of $I_* R$ are derivable in $T$ then so is its conclusion.

$$\textsc{TT-Subst}\quad \frac{\Theta;\ \Gamma \vdash \{x{:}A\}\ \mathcal{J} \qquad \Theta;\ \Gamma \vdash t : A}{\Theta;\ \Gamma \vdash \mathcal{J}[t/x]} \qquad \textsc{TT-Bdry-Subst}\quad \frac{\Theta;\ \Gamma \vdash \{x{:}A\}\ \mathcal{B} \qquad \Theta;\ \Gamma \vdash t : A}{\Theta;\ \Gamma \vdash \mathcal{B}[t/x]}$$

$$\textsc{TT-Subst-EqTy}\quad \frac{\Theta;\ \Gamma \vdash \{x{:}A\}\{\vec{y}{:}\vec{B}\}\ C\ \mathsf{type} \qquad \Theta;\ \Gamma \vdash s : A \qquad \Theta;\ \Gamma \vdash t : A \qquad \Theta;\ \Gamma \vdash s \equiv t : A}{\Theta;\ \Gamma \vdash \{\vec{y}{:}\vec{B}[s/x]\}\ C[s/x] \equiv C[t/x]}$$

$$\textsc{TT-Subst-EqTm}\quad \frac{\Theta;\ \Gamma \vdash \{x{:}A\}\{\vec{y}{:}\vec{B}\}\ u : C \qquad \Theta;\ \Gamma \vdash s : A \qquad \Theta;\ \Gamma \vdash t : A \qquad \Theta;\ \Gamma \vdash s \equiv t : A}{\Theta;\ \Gamma \vdash \{\vec{y}{:}\vec{B}[s/x]\}\ u[s/x] \equiv u[t/x] : C[s/x]}$$

$$\textsc{TT-Conv-Abstr}\quad \frac{\Theta;\ \Gamma \vdash \{x{:}A\}\ \mathcal{J} \qquad \Theta;\ \Gamma \vdash B\ \mathsf{type} \qquad \Theta;\ \Gamma \vdash A \equiv B}{\Theta;\ \Gamma \vdash \{x{:}B\}\ \mathcal{J}}$$

**Fig. 8** Admissible substitution rules

## 2.4 Finitary Rules and Type Theories

Raw rules are *syntactically* well-behaved: the premises and the conclusion are syntactically well-formed entities, and all metavariables, free variable and bound variables well-scoped. Nevertheless, a raw rule may be ill-formed for type-theoretic reasons, a deficiency rectified by the next definition.

Recall that a ***well-founded order*** on a set $I$ is an irreflexive and transitive relation $\prec$ satisfying, for each $S \subseteq I$,

$$(\forall i \in I\,.\,(\forall j \prec i\,.\,j \in S) \Rightarrow i \in S) \Rightarrow S = I.$$

The logical reading of the above condition is an induction principle: in order to show $\forall x \in I\,.\,\phi(x)$ one has to prove, for any $i \in I$, that $\phi(i)$ holds assuming that $\phi(j)$ does for all $j \prec i$.

**Definition 2.23** Given a raw theory $T$ over a symbol signature $\Sigma$, a raw rule $\Theta \Longrightarrow \mathcal{b}\boxed{e}$ over $\Sigma$ is ***finitary*** over $T$ when $\vdash_T \Theta\ \mathsf{mctx}$ and $\Theta;\ [\,] \vdash_T \mathcal{b}$. Similarly, a raw rule-boundary $\Theta \Longrightarrow \mathcal{b}$ is finitary when $\vdash_T \Theta\ \mathsf{mctx}$ and $\Theta;\ [\,] \vdash_T \mathcal{b}$.

A ***finitary type theory*** is a raw type theory $(R_i)_{i\in I}$ for which there exists a well-founded order $(I, \prec)$ such that each $R_i$ is finitary over $(R_j)_{j\prec i}$.

The type theories with context in this paper correspond loosely to the fragment of general type theories [6] where the arities of symbols and rules are restricted to be finite, while general type theories allow the premises to be families of arbitrary size. While raw type theories are already subject to this restriction, we reserve the name *finitary* for the "good" rules and theories, that are well-formed according to the above definition.

Examples of rules that exhibit problematic circularities which are ruled out by the *finitary* requirements can be found in the section on "Acceptable type theories" in [6]; see also Sect. 6 of *loc. cit.* for a thorough discussion of the merits of well-founded presentations of type theories.

***Example 2.24*** We take stock by considering several examples of rules. The rule

$$\textsc{Unique- Ty}\ \frac{\vdash \mathsf{A}\ \mathsf{type} \quad \vdash \mathsf{B}\ \mathsf{type} \quad \vdash \mathsf{t} : \mathsf{A}\ \mathsf{type} \quad \vdash \mathsf{t} : \mathsf{B}\ \mathsf{type}}{\vdash \mathsf{A} \equiv \mathsf{B}}$$

is not raw because it introduces the metavariable t twice, and hence gives rise to a syntactically ill-formed metavariable context. Assuming Π has arity $(\mathsf{Ty}, [(\mathsf{Ty}, 0), (\mathsf{Ty}, 1)])$, consider the rules

$$\textsc{Ty- Π- Short}\ \frac{\vdash \{x{:}\mathsf{A}\}\ \mathsf{B}(x)\ \mathsf{type}}{\vdash \Pi(\mathsf{A}, \{x\}\mathsf{B}(x))} \qquad \textsc{Ty- Π- Long}\ \frac{\vdash \mathsf{A}\ \mathsf{type} \quad \vdash \{x{:}\mathsf{A}\}\ \mathsf{B}(x)\ \mathsf{type}}{\vdash \Pi(\mathsf{A}, \{x\}\mathsf{B}(x))}$$

The rule Ty-Π-Short is not raw because it fails to introduce the metavariable A, while Ty-Π-Long is finitary over any theory. The rule

$$\textsc{Succ- Congr- Typo}\ \frac{\vdash \mathsf{m} : \mathsf{nat} \quad \vdash \mathsf{n} : \mathsf{bool} \quad \vdash \mathsf{m} \equiv \mathsf{n} : \mathsf{nat}}{\vdash \mathsf{succ(m)} \equiv \mathsf{succ(n)} : \mathsf{nat}}$$

is raw when the symbols bool, nat, and succ respectively have arities (Ty, []), (Ty, []), and (Tm, [(Tm, 0)]). Whether it is also finitary depends on a theory. For instance, given the raw rules

$$\textsc{Ty- Bool}\ \frac{}{\vdash \mathsf{bool}\ \mathsf{type}} \qquad \textsc{Ty- Nat}\ \frac{}{\vdash \mathsf{nat}\ \mathsf{type}} \qquad \textsc{Tm- Succ}\ \frac{\vdash \mathsf{n} : \mathsf{nat}}{\vdash \mathsf{succ(n)} : \mathsf{nat}} \qquad \textsc{Bool- Eq- Nat}\ \frac{}{\vdash \mathsf{bool} \equiv \mathsf{nat}}$$

the rule Succ- Congr- Typo is not finitary over the first three rules, but is finitary over all four of them. As a last example, given the symbol Id with arity $(\mathsf{Ty}, [(\mathsf{Ty}, 0), (\mathsf{Tm}, 0), (\mathsf{Tm}, 0)])$, the rules

$$\textsc{Ty- Id}\ \frac{\vdash \mathsf{A}\ \mathsf{type} \quad \vdash \mathsf{s} : \mathsf{A} \quad \vdash \mathsf{t} : \mathsf{A}}{\vdash \mathsf{Id(A, s, t)}\ \mathsf{type}} \qquad \textsc{Ty- Id- Typo}\ \frac{\vdash \mathsf{A}\ \mathsf{type} \quad \vdash \mathsf{s} : \mathsf{A} \quad \vdash \mathsf{t} : \mathsf{A}}{\vdash \mathsf{Id(A, s, s)}\ \mathsf{type}}$$

$$\textsc{Eq- Reflect}\ \frac{\vdash \mathsf{A}\ \mathsf{type} \quad \vdash \mathsf{s} : \mathsf{A} \quad \vdash \mathsf{t} : \mathsf{A} \quad \vdash \mathsf{p} : \mathsf{Id(A, s, t)}}{\vdash \mathsf{s} \equiv \mathsf{t} : \mathsf{A}}$$

are all raw, both Ty- Id and Ty- Id- Typo are finitary over an empty theory, while Eq- Reflect is finitary over a theory containing Ty- Id. The rule Ty- Id is a symbol rule, but Ty- Id- Typo is not.

Could we have folded Definition 2.5 of raw rules and Definition 2.23 of finitary rules into a single definition? Not easily, as that would generate a loop: finitary rules refer to theories and derivability, which refer to closure rules, which are generated from raw rules. Without a doubt something is to be learned by transforming the cyclic dependency to an inductive definition, but we do not attempt to do so here.

A finitary type theory is fairly well behaved from a type-theoretic point of view, but can still suffer from unusual finitary rules, such as Ty- Id- Typo from Example 2.24, which looks

like a spelling mistake. We thus impose a further restriction by requiring that every rule be either a symbol rule or an equality rule.

**Definition 2.25** A finitary type theory is ***standard*** if its specific object rules are symbol rules, and each symbol has precisely one associated rule.

A standard type theory and its symbol signature may be built iteratively as follows:

1. The empty theory is standard over the empty signature.
2. Given a standard type theory $T$ over $\Sigma$, and a rule-boundary

$$\mathsf{M}_1{:}\mathscr{B}_1, \ldots, \mathsf{M}_n{:}\mathscr{B}_n \Longrightarrow \mathscr{b}$$

finitary for $T$:

- If $\mathscr{b}$ is an object boundary, and $\mathsf{S} \notin |\Sigma|$, then $T$ extended with the associated symbol rule

$$\mathsf{M}_1{:}\mathscr{B}_1, \ldots, \mathsf{M}_n{:}\mathscr{B}_n \Longrightarrow \mathscr{b}\boxed{\mathsf{S}(\widehat{\mathsf{M}}_1, \ldots, \widehat{\mathsf{M}}_n)}$$

  is standard over the extended signature $\langle \Sigma, \mathsf{S} \mapsto \alpha \rangle$, where $\alpha$ is the symbol arity associated with the rule-boundary.
- If $\mathscr{b}$ is an equation boundary, then $T$ extended with the equality rule

$$\mathsf{M}_1{:}\mathscr{B}_1, \ldots, \mathsf{M}_n{:}\mathscr{B}_n \Longrightarrow \mathscr{b}\boxed{\star}$$

  is standard over $\Sigma$.

A more elaborate well-founded induction may be employed when a theory features infinitely many rules, such as an infinite succession of universes.

## 3 Meta-theorems

We put our definitions to the test by proving meta-theorems which stipulate desirable structural properties of type theories. The theorems are all rather standard and expected. Nevertheless, we prove them to verify that our definition of type theories is sensible, and to provide general-purpose meta-theorems that apply in a wide range of situations.

Making the statements precise in full generality has not always been trivial. We therefore include them here, together with statements of auxiliary lemmas, to give the reader an overview of the technique, but mostly relegate the rather lengthy induction proofs to the appendix. We shall continue to do so in subsequent sections.

### 3.1 Meta-theorems About Raw Theories

A ***renaming*** of an expression $u$ is an injective map $\rho$ with domain $\mathsf{mv}(u) \cup \mathsf{fv}(u)$ that takes metavariables to metavariables and free variables to free variables. The renaming acts on $u$ to yield an expression $\rho_* u$ by replacing each occurrence of a metavariable $\mathsf{M}$ and a free variable $\mathsf{a}$ with $\rho(\mathsf{M})$ and $\rho(\mathsf{a})$, respectively. We similarly define renamings of contexts, judgements, and boundaries.

**Proposition 3.1** (Renaming) *If a raw type theory derives a judgement or a boundary, then it also derives its renamings.*

***Proof*** Let $\rho$ be a renaming of a derivable judgement $\Theta; \Gamma \vdash \mathcal{J}$. We show that $\rho_*\Theta; \rho_*\Gamma \vdash \rho_*\mathcal{J}$ is derivable by induction on the derivation. The case of boundaries is similar.

Most cases only require a direct application of the induction hypotheses to the premises. The only somewhat interesting case is TT- Abstr,

$$\frac{\Theta; \Gamma \vdash A\ \mathsf{type} \qquad \mathsf{a} \notin |\Gamma| \qquad \Theta; \Gamma, \mathsf{a}{:}A \vdash \mathcal{J}[\mathsf{a}/x]}{\Theta; \Gamma \vdash \{x{:}A\}\ \mathcal{J}}$$

As $\mathsf{a} \notin |\Gamma|$, and thus $\mathsf{a} \notin |\rho|$, we may extend $\rho$ to a renaming $\rho' = \langle \rho, \mathsf{a} \mapsto \mathsf{b} \rangle$, where $\mathsf{b}$ is such that $\mathsf{b} \notin |\rho_*\Gamma|$. By induction hypothesis for the first premise, $\rho_*\Theta; \rho_*\Gamma \vdash \rho_*A\ \mathsf{type}$ is derivable. We apply the induction hypothesis for the second premise to $\rho'$ and obtain $\rho'_*\Theta; \rho'_*(\Gamma, \mathsf{a}{:}A) \vdash \rho'_*(\mathcal{J}[\mathsf{a}/x])$, which equals $\rho_*\Theta; \rho_*\Gamma, \mathsf{b}{:}\rho_*A \vdash (\rho_*\mathcal{J})[\mathsf{b}/x]$. Thus, we may conclude by TT- Abstr,

$$\frac{\rho_*\Theta; \rho_*\Gamma \vdash (\rho_*A)\ \mathsf{type} \qquad \mathsf{b} \notin |\rho_*\Gamma| \qquad \rho_*\Theta; \rho_*\Gamma, \mathsf{b}{:}\rho_*A \vdash (\rho_*\mathcal{J})[\mathsf{b}/x]}{\rho_*\Theta; \rho_*\Gamma \vdash \{x{:}\rho_*A\}\ \rho_*\mathcal{J}}$$

□

**Proposition 3.2** (Weakening) *For a raw type theory:*

1. *If* $\Theta; \Gamma_1, \Gamma_2 \vdash \mathcal{J}$ *and* $\mathsf{a} \notin |\Gamma_1, \Gamma_2|$ *then* $\Theta; \Gamma_1, \mathsf{a}{:}A, \Gamma_2 \vdash \mathcal{J}$.
2. *If* $\Theta_1, \Theta_2; \Gamma \vdash \mathcal{J}$ *and* $\mathsf{M} \notin |\Theta_1, \Theta_2|$ *then* $\Theta_1, \mathsf{M}{:}\mathcal{B}, \Theta_2; \Gamma \vdash \mathcal{J}$.

*An analogous statement holds for boundaries.*

***Proof*** Once again we proceed by induction on the derivation of the judgement in a straightforward manner, where the case TT- Abstr relies on renaming (Proposition 3.1) to ensure that $\mathsf{a}$ remains fresh in the subderivations. □

In several places we shall require well-formedness of contexts, a useful consequence of which we record first.

**Proposition 3.3** *If a raw type theory derives* $\vdash \Theta\ \mathsf{mctx}$ *then it derives* $\Theta; [\,] \vdash \Theta(\mathsf{M})$ *for every* $\mathsf{M} \in |\Theta|$*; and if it derives* $\Theta \vdash \Gamma\ \mathsf{vctx}$*, then it derives* $\Theta; \Gamma \vdash \Gamma(\mathsf{a})\ \mathsf{type}$ *for every* $\mathsf{a} \in |\Gamma|$.

***Proof*** By induction on the derivation of $\vdash \Theta\ \mathsf{mctx}$ and $\Theta \vdash \Gamma\ \mathsf{vctx}$, respectively, followed by weakening. □

### 3.1.1 Admissibility of Substitution

In this section we prove that in a raw type theory substitution rules are derivable closure rules in the sense of Sect. 2.2, and that substitution preserves judgemental equality.

**Lemma 3.4** *If a raw type theory derives* $\Theta; \Gamma, \mathsf{a}{:}A, \Delta \vdash \mathcal{J}$ *and* $\Theta; \Gamma \vdash t : A$ *then it derives* $\Theta; \Gamma, \Delta[t/\mathsf{a}] \vdash \mathcal{J}[t/\mathsf{a}]$.

***Proof*** See the proof on Page 64. □

**Lemma 3.5** *If a raw type theory derives* $\Theta; \Gamma, \mathsf{a}{:}A, \Delta \vdash \mathcal{B}$ *and* $\Theta; \Gamma \vdash t : A$ *then it derives* $\Theta; \Gamma, \Delta[t/\mathsf{a}] \vdash \mathcal{B}[t/\mathsf{a}]$.

***Proof*** The base cases immediately reduce to the previous lemma. The case of TT- BDRY- ABSTR is similar to the case of TT- ABSTR in the previous lemma. □

**Lemma 3.6** *In a raw type theory the following closure rules are admissible:*

$$\text{TT- SUBST}\quad \frac{\Theta;\Gamma \vdash \{x{:}A\}\ \mathcal{J} \qquad \Theta;\Gamma \vdash t : A}{\Theta;\Gamma \vdash \mathcal{J}[t/x]} \qquad \text{TT- BDRY- SUBST}\quad \frac{\Theta;\Gamma \vdash \{x{:}A\}\ \mathcal{B} \qquad \Theta;\Gamma \vdash t : A}{\Theta;\Gamma \vdash \mathcal{B}[t/x]}$$

$$\text{TT- CONV- ABSTR}\quad \frac{\Theta;\Gamma \vdash \{x{:}A\}\ \mathcal{J} \qquad \Theta;\Gamma \vdash B\ \mathsf{type} \qquad \Theta;\Gamma \vdash A \equiv B}{\Theta;\Gamma \vdash \{x{:}B\}\ \mathcal{J}}$$

***Proof*** See the proof on Page 66. □

The next lemma claims that substitution preserves equality, but is a bit finicky to state. Given terms $s$ and $t$, and an object judgement $\mathcal{J}$, define $\mathcal{J}[(s \equiv t)/\mathsf{a}]$ by

$$\begin{aligned}(A\ \mathsf{type})[(s \equiv t)/\mathsf{a}] &= (A[s/\mathsf{a}] \equiv A[t/\mathsf{a}])\\ (u : A)[(s \equiv t)/\mathsf{a}] &= (u[s/\mathsf{a}] \equiv u[t/\mathsf{a}] : A[s/\mathsf{a}])\\ (\{x{:}A\}\ \mathcal{J})[(s \equiv t)/\mathsf{a}] &= (\{x{:}A[s/\mathsf{a}]\}\ \mathcal{J}[(s \equiv t)/\mathsf{a}]).\end{aligned}$$

That is, $\mathcal{J}[(s \equiv t)/\mathsf{a}]$ descends into abstractions by substituting $s$ for $\mathsf{a}$ in the types, and distributes types and terms over the equation $s \equiv t$.

**Lemma 3.7** *If a raw type theory derives*

$$\Theta;\Gamma \vdash s : A, \tag{3.1}$$
$$\Theta;\Gamma \vdash t : A, \tag{3.2}$$
$$\Theta;\Gamma \vdash s \equiv t : A. \tag{3.3}$$
$$\Theta;\Gamma, \mathsf{a}{:}A, \Delta \vdash \mathcal{J}, \tag{3.4}$$
$$\Theta;\Gamma, \Delta[s/\mathsf{a}] \vdash B[s/\mathsf{a}] \equiv B[t/\mathsf{a}] \quad \textit{for all } \mathsf{b} \in |\Delta| \textit{with } \Delta(\mathsf{b}) = B, \tag{3.5}$$

*then it derives*

1. $\Theta;\Gamma, \Delta[s/\mathsf{a}] \vdash \mathcal{J}[s/\mathsf{a}]$,
2. $\Theta;\Gamma, \Delta[s/\mathsf{a}] \vdash \mathcal{J}[t/\mathsf{a}]$, *and*
3. $\Theta;\Gamma, \Delta[s/\mathsf{a}] \vdash \mathcal{J}[(s \equiv t)/\mathsf{a}]$ *if* $\mathcal{J}$ *is an object judgement.*

***Proof*** See the proof on Page 66. □

**Theorem 3.8** (Admissibility of substitution) *In a raw type theory, the closure rules from Fig. 8 are admissible.*

***Proof*** We already established admissibility of TT- SUBST, TT- BDRY- SUBST, and TT- - CONV- ABSTR in Lemma 3.6. Both TT- SUBST- EQTY and TT- SUBST- EQTM are seen to be admissible the same way: invert the abstraction and apply Lemma 3.7 to derive the desired conclusion. □

We provide two more lemmas that allow us to combine substitutions and judegmental equalities more flexibly.

**Lemma 3.9** *Suppose a raw type theory derives*

$$\Theta; \Gamma \vdash s : A, \qquad \Theta; \Gamma \vdash t : A, \quad \textit{and} \quad \Theta; \Gamma \vdash s \equiv t : A.$$

1. *If it derives*

$$\Theta; \Gamma \vdash \{x{:}A\}\{\vec{y}{:}\vec{B}\}\ C \equiv D \quad \textit{and} \quad \Theta; \Gamma \vdash \{x{:}A\}\{\vec{y}{:}\vec{B}\}\ D\ \mathsf{type}$$

   *then it derives* $\Theta; \Gamma \vdash \{\vec{y}{:}\vec{B}[s/x]\}\ C[s/x] \equiv D[t/x]$.
2. *If it derives*

$$\Theta; \Gamma \vdash \{x{:}A\}\{\vec{y}{:}\vec{B}\}\ u \equiv v : C \quad \textit{and} \quad \Theta; \Gamma \vdash \{x{:}A\}\{\vec{y}{:}\vec{B}\}\ v : C$$

   *then it derives* $\Theta; \Gamma \vdash \{\vec{y}{:}\vec{B}[s/x]\}\ u[s/x] \equiv v[t/x] : C[s/x]$.

***Proof*** See the proof on Page 69. □

**Lemma 3.10** *Suppose a raw type theory derives, for* $i = 1, \ldots, n$,

$$\Theta; \Gamma \vdash s_i : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}]$$
$$\Theta; \Gamma \vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}]$$
$$\Theta; \Gamma \vdash s_i \equiv t_i : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}].$$

*If it derives an object judgement* $\Theta; \Gamma \vdash \{\vec{x}{:}\vec{A}\}\ \mathcal{B}\boxed{e}$ *then it derives*

$$\Theta; \Gamma \vdash (\mathcal{B}[\vec{s}/\vec{x}])\boxed{e[\vec{s}/\vec{x}] \equiv e[\vec{t}/\vec{x}]}.$$

***Proof*** See the proof on Page 69. □

### 3.1.2 Admissibility of Instantiations

We next turn to admissibility of instantiations, i.e. preservation of derivability under instantiation of metavariables by heads of derivable judgements.

**Definition 3.11** An instantiation $I = \langle \mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_n \mapsto e_n \rangle$ of a metavariable context $\Xi = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$ over $\Theta; \Gamma$ is ***derivable*** when $\Theta; \Gamma \vdash (I_{(k)*}\mathcal{B}_k)\boxed{e_k}$ is derivable for $k = 1, \ldots, n$.

**Lemma 3.12** *In a raw type theory, let* $I$ *be a derivable instantiation of* $\Xi$ *over context* $\Theta; \Gamma$. *If* $\Xi; \Gamma, \Delta \vdash \mathcal{J}$ *is derivable then so is* $\Theta; \Gamma, I_*\Delta \vdash I_*\mathcal{J}$, *and similarly for boundaries.*

***Proof*** See the proof on Page 70. □

**Theorem 3.13** (Admissibility of instantiation) *In a raw type theory, let* $I$ *be a derivable instantiation of* $\Xi$ *over context* $\Theta; \Gamma$. *If* $\Xi; \Gamma \vdash \mathcal{J}$ *is derivable then so is* $\Theta; \Gamma \vdash I_*\mathcal{J}$, *and similarly for boundaries.*

***Proof*** Apply Lemma 3.12 with empty $\Delta$. □

We next show that, under favorable conditions, instantiating by judgementally equal instantiations leads to judgemental equality. To make the claim precise, define the notation $(I \equiv J)_*\mathcal{J}$ by

$$(I \equiv J)_*(A\ \mathsf{type}) = (I_*A \equiv J_*A\ \mathsf{by}\ \star),$$
$$(I \equiv J)_*(t : A) = (I_*t \equiv J_*t : I_*A\ \mathsf{by}\ \star),$$
$$(I \equiv J)_*(\{x{:}A\}\ \mathcal{J}) = (\{x{:}I_*A\}\ (I \equiv J)_*\mathcal{J})$$

and say that instantiations

$$I = \langle \mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_n \mapsto e_n \rangle \qquad \text{and} \qquad J = \langle \mathsf{M}_1 \mapsto f_1, \ldots, \mathsf{M}_n \mapsto f_n \rangle$$

of $\Xi = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$ over $\Theta; \Gamma$ are ***judgementally equal*** when, for $k = 1, \ldots, n$, if $\mathcal{B}_k$ is an object boundary then $\Theta; \Gamma \vdash (I_{(k)*}\mathcal{B}_k)\boxed{e_k \equiv f_k}$ is derivable.

**Lemma 3.14** *In a raw type theory, consider derivable instantiations $I$ and $J$ of $\Xi = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$ over $\Theta; \Gamma$ which are judgementally equal. Suppose that $\vdash \Xi\ \mathsf{mctx}$ and $\Theta \vdash \Gamma\ \mathsf{vctx}$, and that $\Theta; \Gamma, \Delta \vdash (I_{(i)*}\mathcal{B}_i)\boxed{J(\mathsf{M}_i)}$ is derivable for $i = 1, \ldots, n$, and additionally that, for all $\mathsf{a} \in |\Delta|$ with $\Delta(\mathsf{a}) = A$, so are*

$$\Theta; \Gamma, I_*\Delta \vdash I_*A\ \mathsf{type},$$
$$\Theta; \Gamma, I_*\Delta \vdash J_*A\ \mathsf{type},$$
$$\Theta; \Gamma, I_*\Delta \vdash I_*A \equiv J_*A$$

*If $\Xi; \Gamma, \Delta \vdash \mathcal{J}$ is derivable then so are*

$$\Theta; \Gamma, I_*\Delta \vdash I_*\mathcal{J}, \tag{3.6}$$

$$\Theta; \Gamma, I_*\Delta \vdash J_*\mathcal{J}, \tag{3.7}$$

$$\Theta; \Gamma, I_*\Delta \vdash (I \equiv J)_*\mathcal{J} \quad \text{if } \mathcal{J} \text{ is an object judgement.} \tag{3.8}$$

***Proof*** See the proof on Page 71. □

Lemma 3.14 imposes conditions on the instantiations and the context which can be reduced to the more familiar assumption of well-typedness of the context, using Lemma 3.14 itself, as follows.

**Lemma 3.15** *In a raw type theory, consider $\Xi = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$ such that $\vdash \Xi\ \mathsf{mctx}$, and derivable instantiations*

$$I = \langle \mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_n \mapsto e_n \rangle \qquad \text{and} \qquad J = \langle \mathsf{M}_1 \mapsto f_1, \ldots, \mathsf{M}_n \mapsto f_n \rangle$$

*of $\Xi$ over $\Theta; \Gamma$ which are judgementally equal. Suppose further that $\Theta \vdash \Gamma\ \mathsf{vctx}$ and $\Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i}$ for $i = 1, \ldots, n$. If $\Theta \vdash (\Gamma, \Delta)\ \mathsf{vctx}$, then for all $\mathsf{a} \in |\Delta|$ with $\Delta(\mathsf{a}) = A$:*

$$\Theta; \Gamma, I_*\Delta \vdash I_*A\ \mathsf{type},$$
$$\Theta; \Gamma, I_*\Delta \vdash J_*A\ \mathsf{type},$$
$$\Theta; \Gamma, I_*\Delta \vdash I_*A \equiv J_*A.$$

***Proof*** See the proof on Page 74. □

**Lemma 3.16** *In a raw type theory, consider $\Xi = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$ such that $\vdash \Xi\ \mathsf{mctx}$, and derivable instantiations*

$$I = \langle \mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_n \mapsto e_n \rangle \qquad \text{and} \qquad J = \langle \mathsf{M}_1 \mapsto f_1, \ldots, \mathsf{M}_n \mapsto f_n \rangle$$

*of $\Xi$ over $\Theta; \Gamma$ which are judgementally equal. Suppose that $\Theta \vdash \Gamma\ \mathsf{vctx}$. Then $\Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i}$ is derivable for $i = 1, \ldots, n$.*

***Proof*** See the proof on Page 75. □

Finally, the lemmas can be assembled into an admissibility theorem about judgementally equal derivable instantiations.

**Theorem 3.17** (Admissibility of instantiation equality) *In a raw type theory, consider derivable instantiations $I$ and $J$ of $\Xi$ over $\Theta; \Gamma$ which are judgementally equal. Suppose that $\vdash \Xi\ \mathsf{mctx}$ and $\Theta \vdash \Gamma\ \mathsf{vctx}$. If an object judgement $\Xi; \Gamma \vdash \mathcal{J}$ is derivable then so is $\Theta; \Gamma \vdash (I \equiv J)_* \mathcal{J}$.*

***Proof*** Lemma 3.14 applies with empty $\Delta$ because the additional precondition for $I$ and $J$ is guaranteed by Lemma 3.16. □

Our last meta-theorem about raw type theories shows that whenever a judgement is derivable, so are its presuppositions, i.e., its boundary is well-formed.

**Theorem 3.18** (Presuppositivity) *If a raw type theory derives $\vdash \Theta\ \mathsf{mctx}$, $\Theta \vdash \Gamma\ \mathsf{vctx}$, and $\Theta; \Gamma \vdash \mathcal{B}\boxed{e}$ then it derives $\Theta; \Gamma \vdash \mathcal{B}$.*

***Proof*** See the proof on Page 76. □

### 3.2 Meta-theorems About Finitary Type Theories

Several closure rules contain premises which at first sight seem extraneous, in particular the boundary premises in rule instantiations (Definition 2.12) and the object premises in a congruence rule (Definition 2.17). While these are needed for raw rules, they ought to be removable for finitary rules, which already have well-formed boundaries. We show that this is indeed the case by providing *economic* versions of the rules, which are admissible in finitary type theories. We also show that the metavariable rules (Definition 2.19) have economic versions that are valid in well-formed metavariable contexts, such as the metavariable contexts of finitary rules. Finitary type theories thus allow us to relegate the verification of boundary premises to the definition of the rules, when finitary conditions are checked once and for all, instead of deriving boundary premises for each instance.

**Proposition 3.19** (Economic version of Definition 2.12) *Let $R$ be the raw rule $\Xi \Longrightarrow \mathcal{b}\boxed{e}$ with $\Xi = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$ such that $\Xi; [\,] \vdash \mathcal{b}$ is derivable, in particular $R$ may be finitary. Then for any instantiation $I = [\mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_n \mapsto e_n]$ over $\Theta; \Gamma$, the following closure rule is admissible:*

$$\frac{\Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{e_i} \quad \text{for } i = 1, \ldots, n}{\Theta; \Gamma \vdash I_*(\mathcal{b}\boxed{e})}\ \textsc{tt- Specific- Eco}$$

***Proof*** To apply $I_*R$, derive the missing premise $\Theta; \Gamma \vdash I_*\mathcal{b}$ via Theorem 3.13. □

**Proposition 3.20** (Economic version of Definition 2.19) *If a raw type theory derives $\vdash \Theta\ \mathsf{mctx}$ and $\Theta \vdash \Gamma\ \mathsf{vctx}$ with $\Theta(\mathsf{M}) = (\{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \mathcal{b})$, the following closure rules are admissible:*

$$\frac{\Theta; \Gamma \vdash t_j : A_j[\vec{t}_{(j)}/\vec{x}_{(j)}] \quad \text{for } j = 1, \ldots, m}{\Theta; \Gamma \vdash (\mathcal{b}[\vec{t}/\vec{x}])\boxed{\mathsf{M}(\vec{t})}}\ \textsc{tt- Meta- Eco}$$

$$\frac{\Theta; \Gamma \vdash s_j \equiv t_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}] \quad \text{for } j = 1, \ldots, m}{\Theta; \Gamma \vdash (\mathcal{b}[\vec{s}/\vec{x}])\boxed{\mathsf{M}_k(\vec{s}) \equiv \mathsf{M}_k(\vec{t})}}\ \textsc{tt- Meta- Congr- Eco}$$

***Proof*** See the proof on Page 77. □

**Lemma 3.21** *In a raw type theory, suppose* $\Xi; \Gamma \vdash \mathcal{B}$*, and consider judgementally equal derivable instantiations* $I, J$ *of* $\Xi$ *over* $\Theta; \Gamma$*. If* $\Theta; \Gamma \vdash (I_*\mathcal{B})\boxed{e}$ *is derivable then so is* $\Theta; \Gamma \vdash (J_*\mathcal{B})\boxed{e}$*.*

***Proof*** See the proof on Page 78. □

**Proposition 3.22** (Economic version of Definition 2.17) *In a finitary type theory, consider one of its object rules* $R$

$$\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n \Longrightarrow \mathcal{b}\boxed{e}.$$

*Given instantiations of its premises,*

$$I = \langle \mathsf{M}_1 \mapsto f_1, \ldots, \mathsf{M}_n \mapsto f_n \rangle \quad \textit{and} \quad J = \langle \mathsf{M}_1 \mapsto g_1, \ldots, \mathsf{M}_n \mapsto g_n \rangle,$$

*over* $\Theta; \Gamma$ *such that* $\vdash \Theta$ mctx *and* $\Theta \vdash \Gamma$ vctx*, the following closure rule is admissible:*

TT- Congr- Eco

$$\frac{\begin{array}{ll}\Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i} & \textit{for equation boundary } \mathcal{B}_i \\ \Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i \equiv g_i} & \textit{for object boundary } \mathcal{B}_i\end{array}}{\Theta; \Gamma \vdash (I_*\mathcal{b})\boxed{I_*e \equiv J_*e}}$$

***Proof*** See the proof on Page 78. □

### 3.3 Meta-theorems About Standard Type Theories

We next investigate to what extent a derivation of a derivable judgement can be reconstructed from the judgement itself. Firstly, a term expression holds enough information to recover a candidate for its type, since a standard type theory associates a unique rule, and thus a unique (type) boundary, to each (term) symbol.

**Definition 3.23** Let $T$ be a standard type theory. The ***natural type*** $\tau_{\Theta;\Gamma}(t)$ of a term expression $t$ with respect to a context $\Theta; \Gamma$ is defined by:

$$\begin{aligned}\tau_{\Theta;\Gamma}(\mathsf{a}) &= \Gamma(\mathsf{a}), \\ \tau_{\Theta;\Gamma}(\mathsf{M}(\vec{t})) &= B[\vec{t}/\vec{x}] && \text{where } \Theta(\mathsf{M}) = (\{\vec{x}{:}\vec{A}\}\ \square : B) \\ \tau_{\Theta;\Gamma}(\mathsf{S}(\vec{e})) &= \langle \vec{\mathsf{M}} \mapsto \vec{e} \rangle_* B && \text{where the (unique) rule for } \mathsf{S} \text{ is } \vec{\mathsf{M}}{:}\vec{\mathcal{B}} \Longrightarrow \square : B\end{aligned}$$

We prove an inversion principle that recovers the "stump" of a derivation of a derivable object judgement.

**Theorem 3.24** (Inversion) *If a standard type theory derives an object judgement then there is a derivation of this judgement which concludes with precisely one of the following rules:*

1. *the variable rule* TT- Var*,*
2. *the metavariable rule* TT- Meta*,*
3. *an instantiation of a symbol rule,*
4. *the abstraction rule* TT- Abstr*,*

5. *the term conversion rule* TT- Conv- Tm *of the form*

$$\frac{\Theta; \Gamma \vdash t : \tau_{\Theta;\Gamma}(t) \qquad \Theta; \Gamma \vdash \tau_{\Theta;\Gamma}(t) \equiv A}{\Theta; \Gamma \vdash t : A}$$

*where* $\tau_{\Theta;\Gamma}(t) \neq A$.

***Proof*** See the proof on Page 79. □

We may keep applying the theorem to all the object premises of a stump to recover the proof-relevant part of the derivation. The remaining proof-irrelevant parts are the equational premises. The inversion theorem yields further desirable meta-theoretic properties of standard type theories.

**Corollary 3.25** *If a standard type theory derives* $\Theta; \Gamma \vdash t : A$ *then it derives* $\Theta; \Gamma \vdash \tau_{\Theta;\Gamma}(t) \equiv A$.

***Proof*** By inversion, $\tau_{\Theta;\Gamma}(t) = A$ or we obtain a derivation of $\vdash \tau_{\Theta;\Gamma}(t) \equiv A$. □

**Theorem 3.26** (Uniqueness of typing) *For a standard type theory:*

1. *If* $\Theta; \Gamma \vdash t : A$ *and* $\Theta; \Gamma \vdash t : B$ *then* $\Theta; \Gamma \vdash A \equiv B$.
2. *If* $\vdash \Theta$ $\mathsf{mctx}$ *and* $\Theta \vdash \Gamma$ $\mathsf{vctx}$ *and* $\Theta; \Gamma \vdash s \equiv t : A$ *and* $\Theta; \Gamma \vdash s \equiv t : B$ *then* $\Theta; \Gamma \vdash A \equiv B$.

***Proof*** The first statement holds because $A$ and $B$ are both judegmentally equal to the natural type of $t$ by Corollary 3.25. The second statement reduces to the first one because the presuppositions $\Theta; \Gamma \vdash t : A$ and $\Theta; \Gamma \vdash t : B$ are derivable by Theorem 3.18. □

## 4 Context-Free Finitary Type Theories

In the forward-chaining style, characteristic of LCF-style theorem provers, which Andromeda 2 is designed to be, a judgement is not construed by reducing a goal to subgoals, but as a value of an abstract datatype, and built by applying an abstract datatype constructor to previously derived judgements. What should such a constructor do when its arguments have mismatching variable contexts? It can try to combine them if possible, or require that the user make sure ahead of time that they match. As was already noted by Geuvers et al. in the context of pure type systems [19], it is best to sidestep the whole issue by dispensing with contexts altogether. In the present section we give a second account of finitary type theories, this time without context and with free variables explicitly annotated with their types. These are actually implemented in the Andromeda 2 trusted nucleus.

Our formulation of ***context-free finitary type theories*** is akin to the $\Gamma_\infty$ formalism for pure type systems [19]. We would like to replace judgements of the form "$\Theta; \Gamma \vdash \mathcal{J}$" with just "$\mathcal{J}$". In traditional accounts of logic, as well as in $\Gamma_\infty$, this is accomplished by explicit type annotations of free variables: rather than having $\mathsf{a} : A$ in the variable context, each occurrence of $\mathsf{a}$ is annotated with its type as $\mathsf{a}^A$.

We use the same idea, although we have to overcome several technical complications, of which the most challenging one is the lack of strengthening, which is the principle stating that if $\Theta; \Gamma, \mathsf{a}{:}A, \Delta \vdash \mathcal{J}$ is derivable and $\mathsf{a}$ does not appear in $\Delta$ and $\mathcal{J}$, then $\Theta; \Gamma, \Delta \vdash \mathcal{J}$ is

derivable. An example of a rule that breaks strengthening for finitary type theories is equality reflection from Example 2.10,

$$\frac{\vdash \mathsf{A}\ \mathsf{type} \qquad \vdash \mathsf{s} : \mathsf{A} \qquad \vdash \mathsf{t} : \mathsf{A} \qquad \vdash \mathsf{p} : \mathsf{Id}(\mathsf{A}, \mathsf{s}, \mathsf{t})}{\vdash \mathsf{s} \equiv \mathsf{t} : \mathsf{A}}$$

Because the conclusion elides the metavariable $\mathsf{p}$, it will not record the fact that a variable may have been used in the derivation of the fourth premise. Consequently, we cannot tell what variables ought to occur in the context just by looking at the judgement thesis. As it turns out, variables elided by derivations of equations are the only culprit, and the situation can be rectified by modifying equality judgements so that they carry additional information about usage of variables. In the present section we show how this is accomplished by revisiting the definition of type theories from Sect. 2 and making the appropriate modifications.

### 4.1 Raw Syntax of Context-Free Type Theories

Apart from removing the variable context and annotating free variables with type expressions, we make three further modifications to the raw syntax: we remove metavariable contexts, and instead annotate metavariables with boundaries; we introduce assumption sets that keep track of variables used in equality derivations; and we introduce explicit conversions.

#### 4.1.1 Free and Bound Variables

The ***bound variables*** $x, y, z, \ldots$ are as before, for example they could be de Bruijn indices, whereas the free variables are annotated explicitly with type expressions. More precisely, given a set of names $\mathsf{a}, \mathsf{b}, \mathsf{c}, \ldots$ a ***free variable*** takes the form $\mathsf{a}^A$ where $A$ is a type expression, cf. Sect. 4.1.3. Two such variables $\mathsf{a}^A$ and $\mathsf{b}^B$ are considered syntactically equal when the symbols $\mathsf{a}$ and $\mathsf{b}$ are the same *and* the type expressions $A$ and $B$ are syntactically equal. Thus it is quite possible to have variables $\mathsf{a}^A$ and $\mathsf{a}^B$ which are different even though $A$ and $B$ are judegmentally equal. In an implementation it may be a good idea to prevent such extravaganza by generating fresh symbols so that each one receives precisely one annotation.

Similarly, metavariables are tagged with boundaries, where again $\mathsf{M}^{\mathcal{B}}$ and $\mathsf{N}^{\mathcal{B}'}$ are considered equal when both the symbols $\mathsf{M}$ and $\mathsf{N}$ are equal and the boundaries $\mathcal{B}$ and $\mathcal{B}'$ are syntactically identical.

#### 4.1.2 Arities and Signatures

Arities of symbols and metavariables are as in Sect. 2.1.2. We keep symbol signatures but eliminate metavariable signature, as their arities are induced by annotations.

#### 4.1.3 Raw Expressions

The raw expressions of a context-free type theory are built over a symbol signature, as summarized in the top part of Fig. 9.

A type expression is either a type symbol $\mathsf{S}$ applied to arguments $e_1, \ldots, e_n$, or a metavariable $\mathsf{M}^{\mathcal{B}}$ applied to term expressions $t_1, \ldots, t_n$ where $\mathsf{ar}(\mathcal{B}) = (\mathsf{Ty}, n)$.

The syntax of term expressions differs from the one in Fig. 1 in two ways. First, we annotate free variables with type expressions and metavariables with boundaries, as was

| | | | |
|---|---|---|---|
| Type expression | $A, B$ | $::= \mathsf{S}(e_1, \ldots, e_n)$ | type symbol application |
| | | $\mid \mathsf{M}^{\mathcal{B}}(t_1, \ldots, t_n)$ | type metavariable application |
| Term expression | $s, t$ | $::= \mathsf{a}^A$ | free variable |
| | | $\mid x$ | bound variable |
| | | $\mid \mathsf{S}(e_1, \ldots, e_n)$ | term symbol application |
| | | $\mid \mathsf{M}^{\mathcal{B}}(t_1, \ldots, t_n)$ | term metavariable application |
| | | $\mid \kappa(t, \alpha)$ | conversion |
| Assumption set | $\alpha, \beta$ | $::= \{\!\{ \ldots, a_i^A, \ldots, x_j, \ldots, \mathsf{M}_k^{\mathcal{B}}, \ldots \}\!\}$ | |
| Argument | $e$ | $::= A$ | type argument |
| | | $\mid t$ | term argument |
| | | $\mid \alpha$ | assumption set |
| | | $\mid \{x\}e$ | abstracted arg. ($x$ bound in $e$) |
| Judgement | $j$ | $::= A\ \mathsf{type}$ | $A$ is a type |
| | | $\mid t : A$ | $t$ has type $T$ |
| | | $\mid A \equiv B\ \mathsf{by}\ \alpha$ | $A$ and $B$ are equal types |
| | | $\mid s \equiv t : A\ \mathsf{by}\ \alpha$ | $s$ and $t$ are equal terms at $A$ |
| Abstracted judgement | $\mathcal{J}$ | $::= j$ | non-abstracted judgement |
| | | $\mid \{x{:}A\}\ \mathcal{J}$ | abstracted jdgt. ($x$ bound in $\mathcal{J}$) |
| Boundary | $\mathcal{b}$ | $::= \square\ \mathsf{type}$ | a type |
| | | $\mid \square : A$ | a term of type $A$ |
| | | $\mid A \equiv B\ \mathsf{by}\ \square$ | type equation boundary |
| | | $\mid s \equiv t : B\ \mathsf{by}\ \square$ | term equation boundary |
| Abstracted boundary | $\mathcal{B}$ | $::= \mathcal{b}$ | non-abstracted boundary |
| | | $\mid \{x{:}A\}\ \mathcal{B}$ | abstracted bdry. ($x$ bound in $\mathcal{B}$) |

**Fig. 9** The raw syntax of context-free finitary type theories

already discussed, where it should be noted that in an annotation $A$ of $\mathsf{a}^A$ or $\mathcal{B}$ of $\mathsf{M}^{\mathcal{B}}$ there may be further free and metavariables, which are also annotated, and so on. We require that a boundary annotation $\mathcal{B}$ be closed with respect to free variables (metavariables may occur). Furthermore, a type annotation $A$ must not contain any "exposed" bound variables, i.e. $A$ should be syntactically valid on its own, without having to appear under an abstraction. Second, we introduce the ***conversion terms*** "$\kappa(t, \alpha)$", which will serve to record the variables used to derive the equality along which $t$ has been converted. The context-free conversion rules CF- Conv- Tm and CF- Conv- EqTm in Sect. 4.2 keep track of the assumptions occurring in derivations of type equalities (along which we convert), by recording them as conversion terms.

The expressions of syntactic classes EqTy and EqTm are the ***assumption sets***, which are finite sets of free and bound variables, and metavariables. As we are already using the curly braces for abstraction, we write finite set comprehension as $\{\!\{ \cdots \}\!\}$. Assumption sets record

the variables and metavariables that are used in a derivation of an equality judgement but may not appear in the boundary of the conclusion.

We ought to be a bit careful about occurrences of variables, since the free variables may occur in variable annotations, and the metavariables in boundary annotations. Figure 10, the context-free analogue of Fig. 2, shows the definitions of free, bound and metavariable occurrences. Note the difference between $\mathsf{fv}_0(e)$, which collects *only* the free variable occurrences not appearing in a type annotation, and $\mathsf{fv}(e)$ which collects them all. Exposed bound variables need not be collected from annotations, as they cannot appear there.

The collection of all free, bound and metavariables occurring in an expression is its ***assumption set*** $\mathsf{asm}(e)$. Sometimes we write $\mathsf{asm}(e_1, \ldots, e_n)$ for the union $\bigcup_i \mathsf{asm}(e_i)$.

### 4.1.4 Substitution and Syntactic Equality

We must review substitution and syntactic equality, because they are affected by annotations, assumption sets, and conversion terms.

There are two kinds of substitutions. An ***abstraction*** $e[x/\mathsf{a}^A]$ transforms the free variable $\mathsf{a}^A$ in $e$ to a bound variable $x$, whereas a ***substitution*** $e[s/x]$ replaces the bound variable $x$ with the term $s$. These are shown in Fig. 11. Note that an abstraction $e[x/\mathsf{a}^A]$ is only valid when $\mathsf{a}^A$ does not appear in any type annotation in $e$, $\mathsf{a}^A \notin \mathsf{fvt}(e)$, because type annotations cannot refer to bound variables. Consequently, abstraction of several variables must be carried out in the reverse order of their dependencies. We abbreviate a series of abstractions $((e[x_1/\mathsf{a}_1^{A_1}]) \cdots)[x_n/\mathsf{a}_n^{A_n}]$ as $e[x_1/\mathsf{a}_1^{A_1}, \ldots, x_n/\mathsf{a}_n^{A_n}]$ or just $e[\vec{x}/\vec{\mathsf{a}}_n^{A_n}]$. Similarly, a series of substitutions $((e[s_1/x_1]) \cdots)[s_n/x_n]$ is written $e[s_1/x_1, \ldots, s_n/x_n]$ or just $e[\vec{s}/\vec{x}]$.

Syntactic equality is treated in a standard way, we only have to keep in mind the fact that symbols are considered syntactically equal if the bare symbols are equal *and* their annotations are equal. More interestingly, since conversion terms and assumption sets carry proof-irrelevant information, they should be ignored in certain situations. For this purpose, define the ***erasure*** $\lfloor e \rfloor$ to be the raw expression $e$ with the assumption sets and conversion terms removed:

$$\lfloor \mathsf{a}^A \rfloor = \mathsf{a}^A, \quad \lfloor x \rfloor = x, \quad \lfloor \kappa(t, \alpha) \rfloor = \lfloor t \rfloor, \quad \lfloor \alpha \rfloor = \star, \quad \lfloor \{x\} e \rfloor = \{x\} \lfloor e \rfloor,$$
$$\lfloor \mathsf{S}(e_1, \ldots, e_n) \rfloor = \mathsf{S}(\lfloor e_1 \rfloor, \ldots, \lfloor e_n \rfloor), \quad \lfloor \mathsf{M}^{\mathcal{B}}(t_1, \ldots, t_n) \rfloor = \mathsf{M}^{\mathcal{B}}(\lfloor t_1 \rfloor, \ldots, \lfloor t_n \rfloor).$$

The mapping $e \mapsto \lfloor e \rfloor$ takes the context-free raw syntax of Fig. 9 to the type-theoretic raw syntax of Fig. 1 where the variables $\mathsf{a}^A$ and the metavariables $\mathsf{M}^{\mathcal{B}}$ are construed as atomic symbols, i.e. their annotations are part of the symbol name.

### 4.1.5 Judgements and Boundaries

The lower part of Fig. 9 summarizes the syntax of context-free judgements and boundaries. Apart from not having contexts, type judgements "$A$ type" and term judgements "$t : A$" are as before. Equality judgements are modified to carry assumption sets: a type equality takes the form "$A \equiv B$ by $\alpha$" and a term equality "$s \equiv t : A$ by $\alpha$".

Boundaries do not change, except of course that they have no contexts. The head of a boundary is filled like before, except that assumption sets are used instead of dummy values, see Fig. 12.

**Free variables *not* in typing annotations:**

$$\mathsf{fv}_0(\mathsf{a}^A) = \{\!|\mathsf{a}^A|\!\}, \qquad \mathsf{fv}_0(x) = \{\!|\!|\!\}, \qquad \mathsf{fv}_0(\{x\}e) = \mathsf{fv}_0(e),$$
$$\mathsf{fv}_0(\mathsf{S}(e_1, \dots, e_n)) = \mathsf{fv}_0(e_1) \cup \dots \cup \mathsf{fv}_0(e_n),$$
$$\mathsf{fv}_0(\mathsf{M}^{\mathcal{B}}(t_1, \dots, t_n)) = \mathsf{fv}_0(t_1) \cup \dots \cup \mathsf{fv}_0(t_n),$$
$$\mathsf{fv}_0(\kappa(t, \alpha)) = \mathsf{fv}_0(t) \cup \mathsf{fv}_0(\alpha),$$
$$\mathsf{fv}_0(\alpha) = \{\!|\mathsf{a}^A \mid \mathsf{a}^A \in \alpha|\!\},$$

**Free variables *only* in typing annotations:**

$$\mathsf{fvt}(e) = \bigcup\{\!|\mathsf{fv}(A) \mid \mathsf{a}^A \in \mathsf{fv}_0(e)|\!\}$$

**Free variables:**

$$\mathsf{fv}(e) = \mathsf{fv}_0(e) \cup \mathsf{fvt}(e).$$

**Exposed bound variables:**

$$\mathsf{bv}(\mathsf{a}^A) = \{\!|\!|\!\}, \qquad \mathsf{bv}(x) = \{\!|x|\!\}, \qquad \mathsf{bv}(\{x\}e) = \mathsf{bv}(e) \setminus \{\!|x|\!\},$$
$$\mathsf{bv}(\mathsf{S}(e_1, \dots, e_n)) = \mathsf{bv}(e_1) \cup \dots \cup \mathsf{bv}(e_n),$$
$$\mathsf{bv}(\mathsf{M}^{\mathcal{B}}(t_1, \dots, t_n)) = \mathsf{bv}(t_1) \cup \dots \cup \mathsf{bv}(t_n),$$
$$\mathsf{bv}(\kappa(t, \alpha)) = \mathsf{bv}(t) \cup \mathsf{bv}(\alpha),$$
$$\mathsf{bv}(\alpha) = \{\!|x \mid x \in \alpha|\!\}.$$

**Metavariables:**

$$\mathsf{mv}(\mathsf{a}^A) = \mathsf{mv}(A), \qquad \mathsf{mv}(x) = \{\!|\!|\!\}, \qquad \mathsf{mv}(\{x\}e) = \mathsf{mv}(e),$$
$$\mathsf{mv}(\mathsf{S}(e_1, \dots, e_n)) = \mathsf{mv}(e_1) \cup \dots \cup \mathsf{mv}(e_n),$$
$$\mathsf{mv}(\mathsf{M}^{\mathcal{B}}(t_1, \dots, t_n)) = \{\!|\mathsf{M}^{\mathcal{B}}|\!\} \cup \mathsf{mv}(\mathcal{B}) \cup \mathsf{mv}(t_1) \cup \dots \cup \mathsf{mv}(t_n),$$
$$\mathsf{mv}(\kappa(t, \alpha)) = \mathsf{mv}(t) \cup \mathsf{mv}(\alpha),$$
$$\mathsf{mv}(\alpha) = \{\!|\mathsf{M}^{\mathcal{B}} \mid \mathsf{M}^{\mathcal{B}} \in \alpha|\!\} \cup \bigcup\{\!|\mathsf{mv}(A) \mid \mathsf{a}^A \in \alpha|\!\}$$
$$\cup \bigcup\{\!|\mathsf{mv}(\mathcal{B}) \mid \mathsf{M}^{\mathcal{B}} \in \alpha|\!\}.$$

$$\mathsf{mv}(\square\ \mathsf{type}) = \{\!|\!|\!\}, \qquad \mathsf{mv}(\square : A) = \mathsf{mv}(A),$$
$$\mathsf{mv}(A \equiv B\ \mathsf{by}\ \square) = \mathsf{mv}(A) \cup \mathsf{mv}(B),$$
$$\mathsf{mv}(s \equiv t : A\ \mathsf{by}\ \square) = \mathsf{mv}(s) \cup \mathsf{mv}(t) \cup \mathsf{mv}(A),$$
$$\mathsf{mv}(\{x{:}A\}\mathcal{B}) = \mathsf{mv}(A) \cup \mathsf{mv}(\mathcal{B}).$$

**Assumption sets:**

$$\mathsf{asm}(e) = \mathsf{fv}(e) \cup \mathsf{bv}(e) \cup \mathsf{mv}(e).$$

**Fig. 10** Context-free variable occurrences and assumption sets

**Abstraction:**

$$\begin{aligned}
\mathsf{a}^A[x/\mathsf{a}^A] &= x, \qquad & y[x/\mathsf{a}^A] &= y,\\
\mathsf{b}^B[x/\mathsf{a}^A] &= \mathsf{b}^B \quad \text{if } \mathsf{a}^A \neq \mathsf{b}^B \text{ and } \mathsf{a}^A \notin \mathsf{fvt}(B)\\
\mathsf{S}(e_1,\dots,e_n)[x/\mathsf{a}^A] &= \mathsf{S}(e_1[x/\mathsf{a}^A],\dots,e_n[x/\mathsf{a}^A]),\\
\mathsf{M}^{\mathcal{B}}(t_1,\dots,t_n)[x/\mathsf{a}^A] &= \mathsf{M}^{\mathcal{B}}(t_1[x/\mathsf{a}^A],\dots,t_n[x/\mathsf{a}^A]),\\
\kappa(t,\alpha)[x/\mathsf{a}^A] &= \kappa(t[x/\mathsf{a}^A],\alpha[x/\mathsf{a}^A]),\\
(\{y\}e)[x/\mathsf{a}^A] &= \{y\}(e[x/\mathsf{a}^A]) \quad \text{if } x \neq y,\\
\alpha[x/\mathsf{a}^A] &= \alpha \quad \text{if } \mathsf{a}^A \notin \alpha,\\
\alpha[x/\mathsf{a}^A] &= (\alpha \setminus \{\!|\mathsf{a}^A|\!\}) \cup \{\!|x|\!\} \quad \text{if } \mathsf{a}^A \in \alpha \text{ and } \mathsf{a}^A \notin \mathsf{fvt}(\alpha),
\end{aligned}$$

**Substitution:**

$$\begin{aligned}
\mathsf{a}^A[s/x] &= \mathsf{a}^A, \qquad x[s/x] = s, \qquad y[s/x] = y \quad \text{if } x \neq y,\\
\mathsf{S}(e_1,\dots,e_n)[s/x] &= \mathsf{S}(e_1[s/x],\dots,e_n[s/x])\\
\mathsf{M}^{\mathcal{B}}(t_1,\dots,t_n)[s/x] &= \mathsf{M}^{\mathcal{B}}(t_1[s/x],\dots,t_n[s/x])\\
\kappa(t,\alpha)[s/x] &= \kappa(t[s/x],\alpha[s/x])\\
(\{y\}e)[s/x] &= \{y\}(e[s/x])\\
\alpha[s/x] &= \alpha \quad \text{if } x \notin \alpha\\
\alpha[s/x] &= (\alpha \setminus \{\!|x|\!\}) \cup \mathsf{asm}(s) \quad \text{if } x \in \alpha.
\end{aligned}$$

**Fig. 11** Abstraction and substitution

**Filling the placeholder with a head:**

$$\begin{aligned}
(\square\ \mathsf{type})\boxed{A} &= (A\ \mathsf{type})\\
(\square : A)\boxed{t} &= (t : A)\\
(A \equiv B\ \mathsf{by}\ \square)\boxed{e} &= (A \equiv B\ \mathsf{by}\ \mathsf{asm}(e))\\
(s \equiv t : A\ \mathsf{by}\ \square)\boxed{e} &= (s \equiv t : A\ \mathsf{by}\ \mathsf{asm}(e))\\
(\{x{:}A\}\ \mathcal{B})\boxed{\{x\}e} &= (\{x{:}A\}\ \mathcal{B}\boxed{e}).
\end{aligned}$$

**Filling the placeholder with an equality:**

$$\begin{aligned}
(\square\ \mathsf{type})\boxed{A_1 \equiv A_2\ \mathsf{by}\ \alpha} &= (A_1 \equiv A_2\ \mathsf{by}\ \alpha),\\
(\square : A)\boxed{t_1 \equiv t_2\ \mathsf{by}\ \alpha} &= (t_1 \equiv t_2 : A\ \mathsf{by}\ \alpha),\\
(\{x{:}A\}\ \mathcal{B})\boxed{e_1 \equiv e_2\ \mathsf{by}\ \alpha} &= (\{x{:}A\}\ \mathcal{B}\boxed{e_1 \equiv e_2\ \mathsf{by}\ \alpha}),
\end{aligned}$$

**Fig. 12** Context-free filling the head of a boundary

Free-variable occurrences in judgements are defined as follows, with $\mathsf{fvt}(\mathcal{J})$ defined analogously to $\mathsf{fvt}(e)$ in Fig. 12:

$$\begin{aligned}
\mathsf{fv}_0(A\ \mathsf{type}) &= \mathsf{fv}_0(A), \qquad \mathsf{fv}_0(t : A) = \mathsf{fv}_0(t) \cup \mathsf{fv}_0(A),\\
\mathsf{fv}_0(A \equiv B\ \mathsf{by}\ \alpha) &= \mathsf{fv}_0(A) \cup \mathsf{fv}_0(B) \cup \mathsf{fv}_0(\alpha),\\
\mathsf{fv}_0(s \equiv t : A\ \mathsf{by}\ \alpha) &= \mathsf{fv}_0(s) \cup \mathsf{fv}_0(t) \cup \mathsf{fv}_0(A) \cup \mathsf{fv}_0(\alpha),\\
\mathsf{fv}_0(\{x : A\}\mathcal{J}) &= \mathsf{fv}_0(A) \cup \mathsf{fv}_0(\mathcal{J}),\\
\mathsf{fv}(\mathcal{J}) &= \mathsf{fv}_0(\mathcal{J}) \cup \mathsf{fvt}(\mathcal{J}).
\end{aligned}$$

$$
\begin{aligned}
I_* x = x, \qquad & I_*(\mathsf{\kappa}(t, \alpha)) = \mathsf{\kappa}(I_* t, I_* \alpha), \\
I_* \mathsf{a}^A = \mathsf{a}^{I_* A}, \qquad & I_*(\{x\} e) = \{x\}(I_* e), \\
I_*(\mathsf{S}(e_1, \ldots, e_k)) &= \mathsf{S}(I_* e_1, \ldots, I_* e_k), \\
I_*(\mathsf{M}_i^{\mathcal{B}_i}(t_1, \ldots, t_{m_i})) &= I(\mathsf{M}_i)[(I_* t_1)/x_1, \ldots, (I_* t_{m_i})/x_{m_i}], \\
I_* \alpha &= \bigcup \{\mathsf{asm}(I(\mathsf{M}_i)) \mid \mathsf{M}_i^{\mathcal{B}_i} \in \alpha\} \\
&\quad \cup \{\!|x \mid x \in \alpha|\!\} \cup \{\!|\mathsf{a}^{I_* A} \mid \mathsf{a}^A \in \alpha|\!\}.
\end{aligned}
$$

$$
\begin{aligned}
I_*(A\ \mathsf{type}) &= (I_* A\ \mathsf{type}), \\
I_*(t : A) &= (I_* t : I_* A), \\
I_*(A \equiv B\ \mathsf{by}\ \alpha) &= (I_* A \equiv I_* B\ \mathsf{by}\ I_* \alpha), \\
I_*(\{x{:}A\}\mathcal{J}) &= \{x{:}I_* A\} I_* \mathcal{J}, \\
I_* \square &= \square.
\end{aligned}
$$

**Fig. 13** The action of a metavariable instantiation

We trust the reader can emulate the above definition to define the set $\mathsf{mv}(\mathcal{J})$ of metavariable occurrences in a judgement $\mathcal{J}$, as well as occurrences of free and metavariables in boundaries.

### 4.1.6 Metavariable Instantiations

Next, let us rethink how metavariable instantiations work in the presence of the newly introduced syntactic constructs. As before an ***instantiation*** is a sequence, representing a map,

$$I = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto e_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto e_n \rangle$$

such that $\mathsf{mv}(\mathcal{B}_i) \subseteq \{\!|\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_{i-1}^{\mathcal{B}_{i-1}}|\!\}$ and $\mathsf{ar}(\mathcal{B}_i) = \mathsf{ar}(e_i)$, for each $i = 1, \ldots, n$. As in Sect. 2.1.5, $I$ acts on an expression $u$, provided that $\mathsf{mv}(u) \subseteq |I|$, by replacing metavariables with the corresponding expressions, see Fig. 13. Note that the action of $I$ on a free variable changes the identity of the variable by acting on its typing annotation.

## 4.2 Context-Free Rules and Type Theories

In this section we adapt the notions of raw and finitary rules and type theories to the context-free setting. We shall be rather telegraphic about it, as the changes are straightforward and require little discussion.

**Definition 4.1** A ***context-free raw rule*** $R$ over a symbol signature $\Sigma$ has the form

$$\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow j$$

where the ***premises*** $\mathcal{B}_i$ and the ***conclusion*** $j$ are closed and syntactically valid over $\Sigma$, $\mathsf{mv}(\mathcal{B}_i) \subseteq \{\!|\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_{i-1}^{\mathcal{B}_{i-1}}|\!\}$ for every $i = 1, \ldots, n$, and $\mathsf{mv}(j) = \{\!|\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n}|\!\}$. We say that $R$ is an ***object rule*** when $j$ is a type or a term judgement, and an ***equality rule*** when $j$ is an equality judgement.

The condition $\mathsf{mv}(j) = \{\!|\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n}|\!\}$ ensures that the conclusion of an instantiation of a raw rule records all uses of variables. We shall need it in the proof of Theorem 6.5.

***Example 4.2*** The context-free version of equality reflection from Example 2.10 is

$$\mathsf{A}^{\square\ \mathsf{type}},\quad \mathsf{s}^{\square:\mathsf{A}^{\square\ \mathsf{type}}},\quad \mathsf{t}^{\square:\mathsf{A}^{\square\ \mathsf{type}}},\quad \mathsf{p}^{\mathsf{Id}(\mathsf{A}^{\square\ \mathsf{type}},\mathsf{s}^{\square:\mathsf{A}^{\square\ \mathsf{type}}},\mathsf{t}^{\square:\mathsf{A}^{\square\ \mathsf{type}}})}$$
$$\Longrightarrow\ \mathsf{s}^{\square:\mathsf{A}^{\square\ \mathsf{type}}} \equiv \mathsf{t}^{\square:\mathsf{A}^{\square\ \mathsf{type}}}\ \mathsf{A}^{\square\ \mathsf{type}}\ \mathsf{by}\ \{\!|\mathsf{p}^{\mathsf{Id}(\mathsf{A}^{\square\ \mathsf{type}},\mathsf{s}^{\square:\mathsf{A}^{\square\ \mathsf{type}}},\mathsf{t}^{\square:\mathsf{A}^{\square\ \mathsf{type}}})}|\!\}$$

which is quite unreadable. We indulge in eliding annotations on any variable that is already typed by a premise or a hypothesis, and write just

$$\text{CF- Eq- Reflect}\quad \frac{\vdash \mathsf{A}\ \mathsf{type} \quad \vdash \mathsf{s} : \mathsf{A} \quad \vdash \mathsf{t} : \mathsf{A} \quad \vdash \mathsf{p} : \mathsf{Id}(\mathsf{A}, \mathsf{s}, \mathsf{t})}{\vdash \mathsf{s} \equiv \mathsf{t} : \mathsf{A}\ \mathsf{by}\ \{\!|\mathsf{p}|\!\}}$$

As there are no contexts, we could remove $\vdash$ too, but we leave it there out of habit. Note how the assumption set in the conclusion must record dependence on $\mathsf{p}$, or else it would violate the assumption set condition of Definition 4.1.

When formulating equality closure rules we face a choice of assumption sets. For example, what should $\gamma$ be in the transitivity rule

$$\frac{\vdash A \equiv B\ \mathsf{by}\ \alpha \qquad \vdash B \equiv C\ \mathsf{by}\ \beta}{\vdash A \equiv C\ \mathsf{by}\ \gamma}\ ?$$

Its intended purpose is to record any assumptions used in the premises but not already recorded by $A$ and $C$, which suggests the requirement

$$\mathsf{asm}(A) \cup \mathsf{asm}(B) \cup \mathsf{asm}(C) \cup \alpha \cup \beta \subseteq \mathsf{asm}(A) \cup \mathsf{asm}(C) \cup \gamma.$$

If we replace $\subseteq$ with $=$ we also avoid any extraneous asumptions, which leads to the following definition.

**Definition 4.3** In a closure rule $([p_1, \ldots, p_n], \mathcal{b}\boxed{\alpha})$ whose conclusion is an equality judgement, $\alpha$ is ***suitable*** when $\mathsf{asm}(p_1, \ldots, p_n) = \mathsf{asm}(\mathcal{b}\boxed{\alpha})$.

Provided that $\mathsf{asm}(\mathcal{b}) \subseteq \mathsf{asm}(p_1, \ldots, p_n)$, we may always take the minimal suitable assumption set $\alpha = \mathsf{asm}(p_1, \ldots, p_n) \setminus \mathsf{asm}(\mathcal{b})$. We do not insist on minimality, even though an implementation might make an effort to keep the assumption sets small, because minimality is not preserved by instantiations, whereas suitability is. We shall indicate the suitability requirement in an equality closure rule by stating it as the side condition "$\alpha$ suitable".

**Definition 4.4** A ***context-free raw rule-boundary*** over a symbol signature $\Sigma$ has the form

$$\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow \mathcal{b}$$

where the boundaries $\mathcal{B}_i$ and $\mathcal{b}$ are closed and syntactically valid over $\Sigma$, $\mathsf{mv}(\mathcal{B}_i) \subseteq \{\!|\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_{i-1}^{\mathcal{B}_{i-1}}|\!\}$ for every $i = 1, \ldots, n$, and $\mathsf{mv}(\mathcal{b}) \subseteq \{\!|\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n}|\!\}$. We say that $R$ is an ***object rule-boundary*** when $\mathcal{b}$ is an object boundary, and an ***equality rule-boundary*** when $\mathcal{b}$ is an equality boundary.

**Definition 4.5** Given an object rule-boundary

$$\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow \mathcal{b}$$

over $\Sigma$, the ***associated symbol arity*** is $(c, [\mathsf{ar}(\mathcal{B}_1), \ldots, \mathsf{ar}(\mathcal{B}_n)])$, where $c \in \{\mathsf{Ty}, \mathsf{Tm}\}$ is the syntactic class of $\mathcal{b}$. The ***associated symbol rule*** for $\mathsf{S} \notin |\Sigma|$ is the raw rule

$$\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow \mathcal{b}\boxed{\mathsf{S}(\widehat{\mathsf{M}}_1^{\mathcal{B}_1}, \ldots, \widehat{\mathsf{M}}_n^{\mathcal{B}_n})}$$

over the extended signature $\langle \Sigma, \mathsf{S} \mapsto (c, [\mathsf{ar}(\mathcal{B}_1), \ldots, \mathsf{ar}(\mathcal{B}_n)])\rangle$, where $\widehat{\mathsf{M}}^{\mathcal{B}}$ is the ***generic application*** of the metavariable $\mathsf{M}^{\mathcal{B}}$, defined as:

1. $\widehat{\mathsf{M}}^{\mathcal{B}} = \{x_1\} \cdots \{x_k\}\mathsf{M}^{\mathcal{B}}(x_1, \ldots, x_k)$ if $\mathsf{ar}(\mathcal{B}) = (c, k)$ and $c \in \{\mathsf{Ty}, \mathsf{Tm}\}$,
2. $\widehat{\mathsf{M}}^{\mathcal{B}} = \{x_1\} \cdots \{x_k\}\{\!|\mathsf{M}^{\mathcal{B}}, x_1, \ldots, x_k|\!\}$ if $\mathsf{ar}(\mathcal{B}) = (c, k)$ and $c \in \{\mathsf{EqTy}, \mathsf{EqTm}\}$.

**Definition 4.6** Given an equality rule-boundary

$$\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow \mathcal{b},$$

the ***associated equality rule*** is

$$\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow \mathcal{b}\boxed{\{\!|\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n}|\!\} \setminus \mathsf{asm}(\mathcal{b})}.$$

**Definition 4.7** An ***instantiation*** of a raw rule

$$R = (\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow \mathcal{b}\boxed{e})$$

over a symbol signature $\Sigma$ is an instantiation $I = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto e_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto e_n\rangle$ of the metavariables of $R$. The ***closure rule*** $I_* R$ associated with $I$ and $R$ is $([p_1, \ldots, p_n, q], r)$ where $p_i$ is $\vdash (I_{(i)*}\mathcal{B}_i)\boxed{e_i}$, $q$ is $\vdash I_*\mathcal{b}$, and $r$ is $\vdash I_*(\mathcal{b}\boxed{e})$.

A minor complication arises when congruence rules (Definition 2.17) are adapted to the context-free setting, because conversions must be inserted. Consider the congruence rule (2.1) for $\Pi$ from Example 2.18. The premise $A_1 \equiv A_2$ ensures that the premise $\{x{:}A_1\}\ B_1(x) \equiv B_2(x)$ is well-formed by conversion of $x$ on the right-hand side from $A_1$ to $A_2$, thus in the context-free version of the rule we should allow for the possibility of an explicit conversion. However, we should not enforce an unnecessary conversion in case $A_1 = A_2$, nor should we require particular conversions, as there may be many ways to convert a term. We therefore formulate flexible congruence rules as follows: if an occurrence of a term $t$ possibly requires conversion, we allow in its place a term $t'$ such that $\lfloor t \rfloor = \lfloor t' \rfloor$.

**Definition 4.8** The ***context-free congruence rules*** associated with a context-free raw type rule

$$\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow A\ \mathsf{type}$$

are closure rules, where

$$I = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto f_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto f_n\rangle, \quad \text{and} \quad J = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto g_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto g_n\rangle,$$

of the following form:

$$\frac{\begin{array}{ll} \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i} & \text{for } i = 1, \ldots, n \\ \vdash (J_{(i)*}\mathcal{B}_i)\boxed{g_i} & \text{for } i = 1, \ldots, n \\ \lfloor g_i' \rfloor = \lfloor g_i \rfloor & \text{for object boundary } \mathcal{B}_i \\ \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i \equiv g_i' \text{ by } \alpha_i} & \text{for object boundary } \mathcal{B}_i \\ & \beta \text{ suitable} \end{array}}{\vdash I_* A \equiv J_* A \text{ by } \beta}$$

Similarly, the congruence rule associated with a raw term rule

$$\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow t : A$$

are closure rules of the form

$$\frac{\begin{array}{ll} \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i} & \text{for } i = 1, \ldots, n \\ \vdash (J_{(i)*}\mathcal{B}_i)\boxed{g_i} & \text{for } i = 1, \ldots, n \\ \lfloor g_i' \rfloor = \lfloor g_i \rfloor & \text{for object boundary } \mathcal{B}_i \\ \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i \equiv g_i' \text{ by } \alpha_i} & \text{for object boundary } \mathcal{B}_i \\ \vdash t' : I_* A & \lfloor t' \rfloor = \lfloor J_* t \rfloor \\ & \beta \text{ suitable} \end{array}}{\vdash I_* t \equiv t' : I_* A \text{ by } \beta}$$

***Example 4.9*** The context-free congruence rules for $\Pi$ from Example 2.18 take the form

$$\frac{\begin{array}{rr} \vdash A_1 \text{ type} & \vdash \{x{:}A_1\}\ B_1 \text{ type} \\ \vdash A_2 \text{ type} & \vdash \{x{:}A_2\}\ B_2 \text{ type} \\ \lfloor A_2' \rfloor = \lfloor A_2 \rfloor & \lfloor \{x\}B_2' \rfloor = \lfloor \{x\}B_2 \rfloor \\ \vdash A_1 \equiv A_2' \text{ by } \alpha_1 & \vdash \{x{:}A_1\}\ B_1 \equiv B_2' \text{ by } \alpha_2 \end{array}}{\vdash \Pi(A_1, \{x\}B_1) \equiv \Pi(A_2, \{x\}B_2) \text{ by } \beta}$$

where the minimal suitable $\beta$ is

$$(\alpha_1 \cup \alpha_2 \cup \mathsf{asm}(A_2', \{x\}B_2')) \setminus (\mathsf{asm}(A_1, A_2, \{x\}B_1, \{x\}B_2)).$$

The type expressions $A_2'$ and $B_2'$ may be chosen in such a way that the equations $\vdash A_1 \equiv A_2'$ by $\alpha_1$ and $\vdash \{x{:}A_1\}\ B_1 \equiv B_2'$ by $\alpha_2$ are well-typed, so long as they match $A_2$ and $B_2$ up to erasure. In this case, we expect to be able to directly use $A_2$ for $A_2'$. The equation $\vdash \{x{:}A_1\}\ B_1 \equiv B_2$ by $\alpha_2$ where we use $B_2$ instead of $B_2'$ is not obviously well-typed, as $B_2$ is a family over $A_2$ rather than $A_1$. Intuitively, $B_2'$ should thus be $B_2$ where uses of $x$ have to first convert along the equation $A_1 \equiv A_2$ by $\alpha_1$.

The context-free metavariable closure rules are in direct analogy with the usual ones from Definition 2.19:

**Definition 4.10** The ***context-free metavariable rules*** associated with the metavariable $\mathsf{M}^{\mathcal{B}}$ where $\mathcal{B} = (\{x_1{:}A_1\} \cdots \{x_n{:}A_n\}\ \mathcal{b})$ are the closure rules

$$\text{CF- Meta}\quad \frac{\begin{array}{ll} \vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] & \text{for } i = 1, \ldots, n \\ \vdash \mathcal{b}[\vec{t}/\vec{x}] & \end{array}}{\vdash (\mathcal{b}[\vec{t}/\vec{x}])\boxed{\mathsf{M}^{\mathcal{B}}(\vec{t})}}$$

where $\vec{x} = (x_1, \ldots, x_n)$, $\vec{t} = (t_1, \ldots, t_n)$. Furthermore, if $\mathcal{b}$ is an object boundary, then the ***metavariable congruence rules*** for $\mathsf{M}^{\mathcal{B}}$ are the closure rules CF- Meta- Congr- Ty and CF- Meta- Congr- Tm displayed in Fig. 14.

The following definition of context-free raw type theories is analogous to Definition 2.21, except that we have to use the context-free versions of structural rules.

**Definition 4.11** A ***context-free raw type theory*** $T$ over a symbol signature $\Sigma$ is a family of context-free raw rules, called the ***specific rules*** of $T$. The ***associated deductive system*** of $T$ consists of:

1. the ***structural rules*** over $\Sigma$:
   (a) the *variable*, *metavariable*, *metavariable congruence*, and *abstraction* closure rules (Fig. 14),
   (b) the *equality* closure rules (Fig. 15),
   (c) the *boundary* closure rules (Fig. 16);
2. the instantiations of the specific rules of $T$ (Definition 4.7);
3. for each specific object rule of $T$, the instantiations of the associated congruence rule (Definition 4.8).

We write $\vdash_T \mathcal{J}$ when $\vdash \mathcal{J}$ is derivable with respect to the deductive system associated to $T$, and similarly for $\vdash_T \mathcal{B}$.

The formulations of the abstraction rules CF- Abstr and CF- Bdry- Abstr are suitable for the backward-chaining style of proof, because their conclusions take a general form. For forward-chaining, we may derive abstraction rules with premises in general form as follows:

CF- Abstr- Fwd

$$\frac{\vdash A\ \mathsf{type} \qquad \vdash \mathcal{J} \qquad \mathsf{a}^A \notin \mathsf{fvt}(\mathcal{J})}{\vdash \{x{:}A\}\ \mathcal{J}[x/\mathsf{a}^A]}$$

CF- Bdry- Abstr- Fwd

$$\frac{\vdash A\ \mathsf{type} \qquad \vdash \mathcal{B} \qquad \mathsf{a}^A \notin \mathsf{fvt}(\mathcal{B})}{\vdash \{x{:}A\}\ \mathcal{B}[x/\mathsf{a}^A]}$$

The side condition $\mathsf{a}^A \notin \mathsf{fvt}(\mathcal{J})$ ensures that $\mathsf{a}^A \notin \mathsf{fv}(\mathcal{J}[x/\mathsf{a}^A])$, hence CF- Abstr- Fwd can be derived as the instance of CF- Abstr

$$\frac{\vdash A\ \mathsf{type} \qquad \mathsf{a}^A \notin \mathsf{fv}(\mathcal{J}[x/\mathsf{a}^A]) \qquad \vdash (\mathcal{J}[x/\mathsf{a}^A])[\mathsf{a}^A/x]}{\vdash \{x{:}A\}\mathcal{J}[x/\mathsf{a}^A]}$$

and similarly for boundary abstractions.

The context-free analogues of the auxiliary judgements $\vdash \Theta\ \mathsf{mctx}$ and $\Theta \vdash \Gamma\ \mathsf{vctx}$ are as follows. For simplicity we define a single notion that encompasses the well-formedness of all annotations.

**Definition 4.12** An expression $u$ has ***well-typed annotations*** when $\vdash \mathcal{B}$ for every $\mathsf{M}^{\mathcal{B}} \in \mathsf{asm}(u)$ and $\vdash A\ \mathsf{type}$ for every $\mathsf{a}^A \in \mathsf{asm}(u)$. The notion evidently extends to judgements and boundaries.

The context-free version of finitary rules and type theories is quite similar to the original one.

**Definition 4.13** Given a raw theory $T$ over a symbol signature $\Sigma$, a context-free raw rule $\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow \mathcal{b}\boxed{e}$ over $\Sigma$ is ***finitary*** over $T$ when $\vdash_T \mathcal{B}_i$ for $k = 1, \ldots, n$, and $\vdash_T \mathcal{b}$, Similarly, a raw rule-boundary $\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow \mathcal{b}$ is finitary over $T$ when $\vdash_T \mathcal{B}_i$ for $k = 1, \ldots, n$, and $\vdash_T \mathcal{b}$.

A ***context-free finitary type theory*** is a context-free raw type theory $(R_i)_{i \in I}$ for which there exists a well-founded order $(I, \prec)$ such that each $R_i$ is finitary over $(R_j)_{j \prec i}$.

**Definition 4.14** A context-free finitary type theory is ***standard*** if its specific object rules are symbol rules, and each symbol has precisely one associated rule.

$$\textsc{CF-Var}\quad \frac{}{\vdash \mathsf{a}^A : A} \qquad \textsc{CF-Abstr}\quad \frac{\vdash A\ \mathsf{type} \qquad \mathsf{a}^A \notin \mathsf{fv}(\mathcal{J}) \qquad \vdash \mathcal{J}[\mathsf{a}^A/x]}{\vdash \{x{:}A\}\ \mathcal{J}}$$

$$\textsc{CF-Meta}\quad \frac{\vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] \quad \text{for } i = 1,\dots,n \qquad \vdash \mathcal{b}[\vec{t}/\vec{x}]}{\vdash (\mathcal{b}[\vec{t}/\vec{x}])\boxed{\mathsf{M}^{\{\vec{x}:\vec{A}\}\mathcal{b}}(\vec{t})}}$$

$$\textsc{CF-Meta-Congr-Ty}\quad \frac{\begin{array}{l}\mathcal{B} = \{x_1{:}A_1\}\cdots\{x_m{:}A_m\}\ \square\ \mathsf{type} \\ \vdash s_j : A[\vec{s}_{(j)}/\vec{x}_{(j)}] \qquad \text{for } j = 1,\dots,m \\ \vdash t_j : A[\vec{t}_{(j)}/\vec{x}_{(j)}] \qquad \text{for } j = 1,\dots,m \\ \lfloor t_j \rfloor = \lfloor t'_j \rfloor \qquad \text{for } j = 1,\dots,m \\ \vdash s_j \equiv t'_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}]\ \mathsf{by}\ \alpha_j \qquad \text{for } j = 1,\dots,m \\ \beta\ \text{suitable}\end{array}}{\vdash \mathsf{M}^{\mathcal{B}}(\vec{s}) \equiv \mathsf{M}^{\mathcal{B}}(\vec{t})\ \mathsf{by}\ \beta}$$

$$\textsc{CF-Meta-Congr-Tm}\quad \frac{\begin{array}{l}\mathcal{B} = \{x_1{:}A_1\}\cdots\{x_m{:}A_m\}\ \square : B \\ \vdash s_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}] \qquad \text{for } j = 1,\dots,m \\ \vdash t_j : A_j[\vec{t}_{(j)}/\vec{x}_{(j)}] \qquad \text{for } j = 1,\dots,m \\ \lfloor t_j \rfloor = \lfloor t'_j \rfloor \qquad \text{for } j = 1,\dots,m \\ \vdash s_j \equiv t'_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}]\ \mathsf{by}\ \alpha_j \qquad \text{for } j = 1,\dots,m \\ \vdash v : B[\vec{s}/\vec{x}] \qquad \lfloor \mathsf{M}^{\mathcal{B}}(\vec{t}) \rfloor = \lfloor v \rfloor \qquad \beta\ \text{suitable}\end{array}}{\vdash \mathsf{M}^{\mathcal{B}}(\vec{s}) \equiv v : B[\vec{s}/\vec{x}]\ \mathsf{by}\ \beta}$$

**Fig. 14** Context-free free variable, metavariable, and abstraction closure rules

## 5 Meta-theorems About Context-Free Theories

The meta-theorems from Sect. 3 carry over to the context-free setting. Unfortunately, there seems to be no wholesale method for transferring the proofs, and one simply has to adapt them manually to the context-free setting. The process is quite straightforward, so we indulge in omitting the details.

### 5.1 Meta-theorems About Context-Free Raw Theories

In the context-free setting, a renaming is still an injective map $\rho$ taking unannotated symbols to unannotated symbols. Its action $\rho_* e$ on an expression $e$ recursively descends into $e$, including

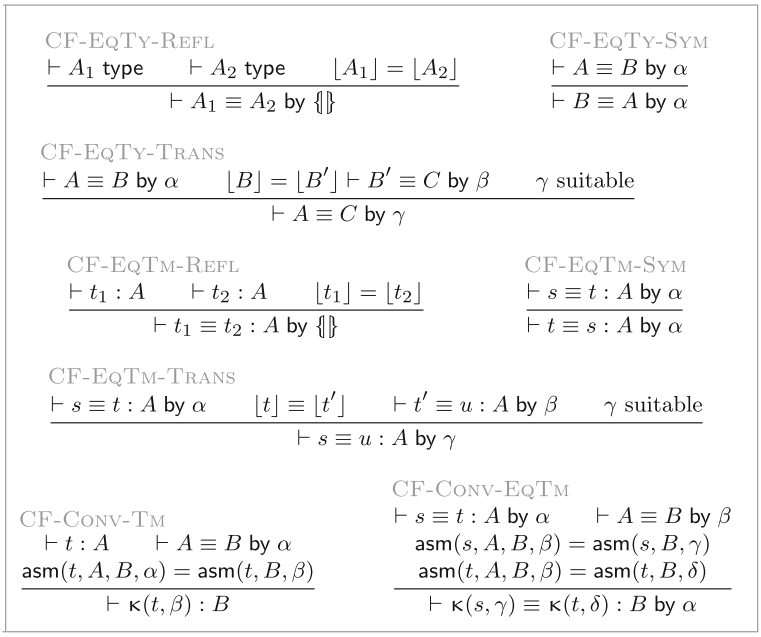


**Fig. 15** Context-free closure rules for equality

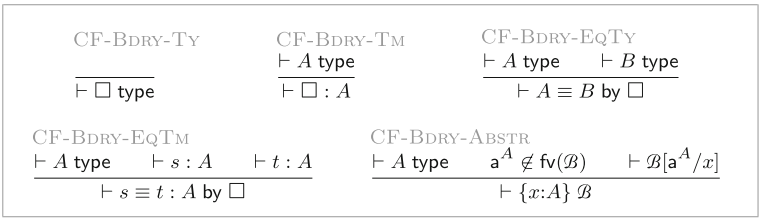


**Fig. 16** Well-formed context-free abstracted boundaries

into variable annotations, i.e. $\rho_*(\mathsf{a}^A) = \rho(\mathsf{a})^{\rho_* A}$ and $\rho_*(\mathsf{M}^{\mathcal{B}}) = \rho(\mathsf{M})^{\rho_* \mathcal{B}}$. The action is extended to judgements and boundaries in a straightforward manner. Renaming preserves the size of an expression, as long as all symbols are deemed to have the same size.

**Proposition 5.1** (Context-free renaming) *If a context-free raw type theory derives a judgement or a boundary, then it also derives its renamings.*

***Proof*** Straightforward induction on the derivation. □

Weakening (Proposition 3.2) is not applicable, as there is no context that could be weakened, and no variable ever occurs in the conclusion of a judgement without it being used in the derivation.

We next prove that substitution rules are admissible closure rules in the sense of Sect. 2.2. We take a slightly different route than in Sect. 3.1 in order to avoid substituting a term for a free variable, as that changes type annotations and therefore the identity of variables. Lemmas 5.2 and 5.3 are proved by mutual structural induction, with a further structural induction within each lemma.

**Lemma 5.2** *If a context-free raw type theory derives*

$$\vdash \{x_1{:}A_1\}\cdots\{x_n{:}A_n\}\ \mathcal{J} \qquad \text{and}$$
$$\vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] \qquad \text{for } i = 1,\ldots,n$$

*then it derives* $\vdash \mathcal{J}[\vec{t}/\vec{x}]$.

***Proof*** See the proof on Page 80. □

**Lemma 5.3** *If a context-free raw type theory derives*

$$\vdash \{x_1{:}A_1\}\cdots\{x_n{:}A_n\}\ \mathcal{B} \qquad \text{and}$$
$$\vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] \qquad \text{for } i = 1,\ldots,n$$

*then it derives* $\vdash \mathcal{B}[\vec{t}/\vec{x}]$.

***Proof*** We proceed as in the proof of Lemma 5.2, where CF- BDRY- ABSTR is treated like CF- ABSTR, and the remaining ones invert to Lemma 5.2. □

**Theorem 5.4** (Context-free admissibility of substitution) *In a context-free raw type theory, the following substitution rules are admissible closure rules:*

$$\text{CF- SUBST}\ \frac{\vdash \{x{:}A\}\ \mathcal{J} \qquad \vdash t : A}{\vdash \mathcal{J}[t/x]} \qquad \text{CF- BDRY- SUBST}\ \frac{\vdash \{x{:}A\}\ \mathcal{B} \qquad \vdash t : A}{\vdash \mathcal{B}[t/x]}$$

***Proof*** The admissibility of CF- SUBST and CF- BDRY- SUBST corresponds to the case $n = 1$ of Lemmas 5.2 and 5.3, respectively. □

Before addressing the context-free versions of TT- SUBST- EQTY and TT- SUBST- EQTM, we prove the context-free presuppositivity theorem.

Of course, presuppositivity holds in the context-free setting as well.

**Theorem 5.5** (Context-free presuppositivity)
*If a context-free raw type theory derives* $\vdash \mathcal{B}\boxed{e}$ *and* $\mathcal{B}\boxed{e}$ *has well-typed annotations, then it derives* $\vdash \mathcal{B}$.

***Proof*** See the proof on Page 81. □

Let us now turn to meta-theorems stating that equal substitutions act equally. Once again we need to account for insertion of conversions. In congruence rules such conversions appeared in premises: equations associated to object premises of the shape $(I_{(i)*}\mathcal{B}_i)\boxed{f_i \equiv g_i' \text{ by } \alpha_i}$ referred to a primed version of $g_i$ to allow the use of conversions in $g_i$. In the following lemma, conversions appear in the result of a substitution. Therefore, rather than being permissive about insertions of conversions, we are faced with showing that it is possible to insert them. Similarly to Lemma 3.7, we prove that equal terms can be substituted into a judgement to yield equal results, but the right hand side of these results is only prescribed up to erasure, namely as $C'$ and $u'$.

**Lemma 5.6** *If a context-free raw type theory derives*

$$\vdash \{x_1{:}A_1\}\cdots\{x_n{:}A_n\}\ \mathcal{J}$$

*where* $\{\vec{x}{:}\vec{A}\}\ \mathcal{J}$ *has well-typed annotations, and for* $i = 1, \ldots, n$

$$\begin{aligned} &\vdash s_i : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}] \\ &\vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] \\ &\vdash s_i \equiv t_i' : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}] \text{ by } \alpha_i \qquad \textit{and } \lfloor t_i' \rfloor = \lfloor t_i \rfloor. \end{aligned} \tag{5.1}$$

*then:*

1. *if* $\mathcal{J} = (\{\vec{y}{:}\vec{B}\}\ C\ \mathsf{type})$ *then there are* $\gamma$ *and* $C'$ *such that* $\lfloor C[\vec{t}/\vec{x}] \rfloor = \lfloor C' \rfloor$,
$$\vdash \{\vec{y}{:}\vec{B}[\vec{s}/\vec{x}]\}\ C[\vec{s}/\vec{x}] \equiv C' \text{ by } \gamma,$$
2. *if* $\mathcal{J} = (\{\vec{y}{:}\vec{B}\}\ u : C)$ *then there are* $\delta$ *and* $u'$ *such that* $\lfloor u[\vec{t}/\vec{x}] \rfloor = \lfloor u' \rfloor$ *and*
$$\vdash \{\vec{y}{:}\vec{B}[\vec{s}/\vec{x}]\}\ u[\vec{s}/\vec{x}] \equiv u' : C[\vec{s}/\vec{x}] \text{ by } \delta.$$

*Furthermore, no extraneous assumptions are introduced by* $\gamma$, $C'$, $\delta$ *and* $u'$*:*

$$\mathsf{asm}(\{\vec{y}\}\gamma, \{\vec{y}\}C', \{\vec{y}\}\delta, \{\vec{y}\}u') \subseteq \mathsf{asm}(\vec{s}, \vec{t}, \vec{t}', \vec{\alpha}, \{\vec{x}{:}\vec{A}\}\ \mathcal{J}).$$

***Proof*** See the proof on Page 83. □

**Theorem 5.7** *In a context-free raw type theory, the following closure rules are admissible:*

CF- Subst- EqTy

$$\frac{\begin{array}{ll} \vdash \{\vec{x}{:}\vec{A}\}\{\vec{y}{:}\vec{B}\}\ C\ \mathsf{type} & \\ \vdash s_i : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}] & \textit{for } i = 1, \ldots, n \\ \vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] & \textit{for } i = 1, \ldots, n \\ \lfloor t_i \rfloor = \lfloor t_i' \rfloor & \textit{for } i = 1, \ldots, n \\ \vdash s_i \equiv t_i' : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}] \text{ by } \alpha_i & \textit{for } i = 1, \ldots, n \\ & \beta \textit{ suitable} \end{array}}{\vdash \{\vec{y}{:}\vec{B}[\vec{s}/\vec{x}]\}\ C[\vec{s}/x] \equiv C[\vec{t}/x] \text{ by } \beta}$$

CF- Subst- EqTm

$$\frac{\begin{array}{ll} \vdash \{\vec{x}{:}\vec{A}\}\{\vec{y}{:}\vec{B}\}\ u : C & \\ \vdash s_i : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}] & \textit{for } i = 1, \ldots, n \\ \vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] & \textit{for } i = 1, \ldots, n \\ \lfloor t_i \rfloor = \lfloor t_i' \rfloor & \textit{for } i = 1, \ldots, n \\ \vdash s_i \equiv t_i' : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}] \text{ by } \alpha_i & \textit{for } i = 1, \ldots, n \\ & \beta \textit{ suitable} \end{array}}{\vdash \{\vec{y}{:}\vec{B}[\vec{s}/\vec{x}]\}\ u[\vec{s}/\vec{x}] \equiv \kappa(u[\vec{t}/\vec{x}], \beta) : C[\vec{s}/x] \text{ by } \beta}$$

***Proof*** See the proof on Page 86. □

Lastly, we prove the context-free counterpart of instantiation admissibility Theorem 3.13. The notion of a derivable instantiation carries over easily to the context-free setting: $I = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto e_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto e_n \rangle$ is ***derivable*** when $\vdash (I_{(i)*}\mathcal{B}_i)\boxed{e_i}$ for every $i = 1, \ldots, n$.

**Theorem 5.8** (Context-free admissibility of instantiation) *In a raw type theory, if* $\vdash \mathcal{J}$ *is derivable, it has well-typed annotations, and* $I$ *is a derivable instantiation such that* $\mathsf{mv}(\mathcal{J}) \subseteq |I|$*, then* $\vdash I_*\mathcal{J}$ *is derivable, and similarly for boundaries.*

***Proof*** See the proof on Page 87. □

### 5.2 Meta-theorems About Context-Free Finitary Theories

The context-free economic rules for finitary theories carry over to the context-free setting. The proofs are analogous to those of Sect. 3.2 so we omit them.

**Proposition 5.9** (Economic version of Definition 4.7) *Let $R$ be the context-free raw rule $\Xi \Longrightarrow \mathcal{b}\boxed{e}$ with $\Xi = [\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n}]$ such that $\vdash \mathcal{b}$ is derivable, in particular $R$ may be finitary. Then for any instantiation $I = [\mathsf{M}_1^{\mathcal{B}_1} \mapsto e_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto e_n]$, the following closure rule is admissible:*

$$\text{CF- Specific- Eco}\quad \frac{\vdash (I_{(i)*}\mathcal{B}_i)\boxed{e_i} \quad \text{for } i = 1, \ldots, n}{\vdash I_*(\mathcal{b}\boxed{e})}$$

**Proposition 5.10** (Economic version of Definition 4.10) *In a context-free raw type theory, if $\mathcal{B} = \{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \mathcal{b}$ and $\mathsf{M}^{\mathcal{B}}$, and $\vec{t}$ have well-typed annotations, then the following closure rule is admissible:*

$$\text{CF- Meta- Eco}\quad \frac{\vdash t_j : A_j[\vec{t}_{(j)}/\vec{x}_{(j)}] \quad \text{for } j = 1, \ldots, m}{\vdash (\mathcal{b}[\vec{t}/\vec{x}])\boxed{\mathsf{M}^{\mathcal{B}}(\vec{t})}}$$

*If, furthermore, $\vec{s}$ has well-typed annotations, then there exists $v$, such that $\lfloor v \rfloor = \lfloor \mathsf{M}^{\mathcal{B}}(\vec{t}) \rfloor$ and the following closure rule is admissible:*

$$\text{CF- Meta- Congr- Eco}\quad \frac{\vdash s_j \equiv t_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}]\ \mathsf{by}\ \alpha_j \quad \text{for } j = 1, \ldots, m \qquad \beta \text{ suitable}}{\vdash (\mathcal{b}[\vec{s}/\vec{x}])\boxed{\mathsf{M}^{\mathcal{B}}(\vec{s}) \equiv v\ \mathsf{by}\ \beta}}$$

### 5.3 Meta-theorems About Context-Free Standard Theories

Inversion and uniqueness of typing (Theorems 3.24, 3.26) carry over to context-free finitary theories. First, the notion of natural type is simpler, as it does not depend on the context anymore.

**Definition 5.11** Let $T$ be a finitary type theory. The *natural type* $\tau(t)$ of a term expression $t$ is defined by:

$$\begin{aligned}
\tau(\mathsf{a}^A) &= A,\\
\tau(\mathsf{M}^{\mathcal{B}}(t_1, \ldots, t_m)) &= A[t_1/x_1, \ldots, t_m/x_m]\\
&\qquad \text{where } \mathcal{B} = (\{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \square : A)\\
\tau(\mathsf{S}(e_1, \ldots, e_n)) &= \langle \mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_n \mapsto e_n \rangle_* B\\
&\qquad \text{where the symbol rule for } \mathsf{S} \text{ is}\\
&\qquad \mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow \square : B\\
\tau(\kappa(t, \alpha)) &= \tau(t)
\end{aligned}$$

Next, we define an operation which peels conversions off a term, and another one that collects the peeled assumption sets. We shall use these in the formulation of the context-free inversion theorem.

**Definition 5.12** The ***conversion-stripping*** $\mathfrak{s}(t)$ of a term expression $t$ is defined by:

$$\mathfrak{s}(t) = \begin{cases} \mathfrak{s}(t') & \text{if } t = \kappa(t', \alpha), \\ t & \text{otherwise.} \end{cases}$$

The ***conversion-residue*** $\mathfrak{r}(t)$ is defined by

$$\mathfrak{r}(t) = \begin{cases} \alpha \cup \mathfrak{r}(t') & \text{if } t = \kappa(t', \alpha), \\ \{\!\} & \text{otherwise.} \end{cases}$$

Note that $\lfloor t \rfloor = \lfloor \mathfrak{s}(t) \rfloor$ and that $\mathsf{asm}(t) = \mathsf{asm}(\mathfrak{s}(t), \mathfrak{r}(t))$.

**Lemma 5.13** *If a context-free standard type theory derives* $\vdash t : A$ *then*

1. *it derives* $\vdash \mathfrak{s}(t) : \tau(t)$ *by an application of* CF- VAR, CF- META, *or an instantiation of a term symbol rule, and*
2. *it derives* $\vdash \tau(t) \equiv A$ by $\mathfrak{r}(t)$.

***Proof*** See the proof on Page 88. □

**Theorem 5.14** (Context-free inversion) *If a context-free standard type theory derives* $\vdash t : A$, *then:*

- *if* $A = \tau(t)$, *it derives* $\vdash \mathfrak{s}(t) : \tau(t)$ *by a derivation which concludes with* CF- VAR, CF- META, *or an instantiation of a term symbol rule;*
- *if* $A \neq \tau(t)$, *it derives* $\vdash \kappa(\mathfrak{s}(t), \mathfrak{r}(t)) : A$ *by* CF- CONV- TM.

***Proof*** Apply Lemma 5.13 and, depending on whether $A = \tau(t)$, either use $\vdash \mathfrak{s}(t) : \tau(t)$ so obtained directly or convert it along $\vdash \tau(t) \equiv A$ by $\mathfrak{r}(t)$, observing that the side condition $\mathsf{asm}(\mathfrak{s}(t), \tau(t), A, \mathfrak{r}(t)) = \mathsf{asm}(\mathfrak{s}(t), \mathfrak{r}(t), A)$ holds because $\mathsf{asm}(\tau(t)) \subseteq \mathsf{asm}(t) = \mathsf{asm}(\mathfrak{s}(t), \mathfrak{r}(t))$. □

**Theorem 5.15** (Context-free uniqueness of typing) *For a context-free standard type theory:*

1. *If* $\vdash t : A$ *and* $\vdash t : B$, *then* $\vdash A \equiv B$ by $\alpha$ *for some assumption set* $\alpha$.
2. *If* $\vdash s \equiv t : A$ by $\beta_1$ *and* $\vdash s \equiv t : B$ by $\beta_2$, *with well-typed variables, then* $\vdash A \equiv B$ by $\alpha$ *for some assumption set* $\alpha$.

*In both cases,* $\alpha \subseteq \mathsf{asm}(t)$ *can be computed from the judgements involved, without recourse to their derivations.*

***Proof*** The first statement holds because $A$ and $B$ are both judegmentally equal to the natural type of $t$ by Lemma 5.13. The second statement reduces to the first one because the presuppositions $\vdash t : A$ and $\vdash t : B$ are derivable by Theorem 5.5. □

## 5.4 Special Meta-theorems About Context-Free Theories

We prove several meta-theorems which are specific to context-free type theories. The example of the equality reflection rule in the beginning of Sect. 4 showcased that finitary type theories do not enjoy strengthening. Context-free type theories, however, do satisfy this meta-property.

**Theorem 5.16** (Strengthening) *If a context-free raw type theory derives*

$$\vdash \{\vec{y}{:}\vec{B}\}\{x{:}A\}\ \mathcal{J}$$

*and* $x \notin \mathsf{bv}(\mathcal{J})$ *then it also derives* $\vdash \{\vec{y}{:}\vec{B}\}\ \mathcal{J}$.

***Proof*** We proceed by induction on the derivation of $\{\vec{y}{:}\vec{B}\}\{x{:}A\}\ \mathcal{J}$. The only case to consider is CF- ABSTR. If the outer abstraction is empty, then the derivation ends with the abstraction

$$\frac{\vdash A\ \mathsf{type} \qquad \mathsf{a}^A \notin \mathsf{fv}(\mathcal{J}) \qquad \vdash \mathcal{J}[\mathsf{a}^A/x]}{\vdash \{x{:}A\}\ \mathcal{J}} \tag{5.2}$$

Because $x \notin \mathsf{bv}(\mathcal{J})$, it follows that $\mathsf{a}^A \notin \mathsf{fv}_0(\mathcal{J}[\mathsf{a}^A/x])$ and that $\mathcal{J}[\mathsf{a}^A/x] = \mathcal{J}$, which is the second premise, hence derivable. The other possibility is that the derivation ends with

$$\frac{\vdash A\ \mathsf{type} \qquad \mathsf{c}^C \notin \mathsf{fv}(\{\vec{y}{:}\vec{B}\}\{x{:}A\}\ \mathcal{J}) \qquad \vdash \{\vec{y}{:}\vec{B}[\mathsf{c}^C/z]\}\{x{:}A[\mathsf{c}^C/z]\}\ \mathcal{J}[\mathsf{c}^C/z]}{\vdash \{z{:}C\}\{\vec{y}{:}\vec{B}\}\{x{:}A\}\ \mathcal{J}}$$

From $x \notin \mathsf{bv}(\mathcal{J})$ it follows that $x \notin \mathsf{bv}(\mathcal{J}[\mathsf{c}^C/z])$, hence we may apply the induction hypothesis to the second premise and conclude by abstracting $\mathsf{c}^C$. □

Why can we not adapt the above proof to type theories with contexts? In the derivation (5.2), the second premise turns out to be precisely the desired conclusion, whereas TT- ABSTR would yield $\Theta;\Gamma, \mathsf{a}{:}A \vdash \mathcal{J}$ where $\Theta;\Gamma \vdash \mathcal{J}$ is needed. Indeed, strengthening is not generally valid for type theories with contexts.

The next lemma can be used to modify the head of a judgement so that it fits another boundary, as long as there is agreement up to erasure.

**Theorem 5.17** (Boundary conversion) *In a context-free raw theory, if* $\vdash \mathcal{B}_1$, $\vdash \mathcal{B}_2$, $\vdash \mathcal{B}_1\boxed{e_1}$ *and* $\lfloor\mathcal{B}_1\rfloor = \lfloor\mathcal{B}_2\rfloor$ *then there is* $e_2$ *such that* $\vdash \mathcal{B}_2\boxed{e_2}$, $\mathsf{asm}(e_2) \subseteq \mathsf{asm}(\mathcal{B}_1\boxed{e_1})$ *and* $\lfloor e_1\rfloor = \lfloor e_2\rfloor$.

***Proof*** See the proof on Page 88. □

## 6 A Correspondence Between Theories With and Without Contexts

We now establish a correpondence between finitary type theories with and without contexts. We use the prefixes "tt" (for "traditional types") and "cf" (for "context-free") to disambiguate between the two versions of type theory. Thus the raw tt-syntax is the one from Fig. 1, and the raw cf-syntax the one from Fig. 9.

To ease the translation between the two versions of type theory, we shall use annotated free variables $\mathsf{a}^A$ and annotated metavariables $\mathsf{M}^{\mathcal{B}}$ in both version of raw syntax, where the annotations $A$ and $\mathcal{B}$ are those of the cf-syntax. In the tt-syntax these annotations are considered part of the symbol names, and do not carry any type-theoretic significance.

### 6.1 Translation from cf-Theories to tt-Theories

We first show how to translate constituents of cf-theories to corresponding constituents of tt-theories. The plan is simple enough: move the annotations to contexts, elide the conversion terms, and replace the assumption sets with the dummy value.

The first step towards the translation was taken in Sect. 4.1.4, where we defined the erasure operation taking a cf-expression $e$ to a tt-expression $\lfloor e\rfloor$ by removing conversions and replacing assumption sets with the dummy value. Note that erasure and substitution commute, $\lfloor e[t/x]\rfloor = \lfloor e\rfloor[\lfloor t\rfloor/x]$, by an induction on the syntactic structure of $e$.

Next, in order to translate cf-judgements to tt-judgements, we need to specify when a context correctly encodes the information provided by cf-annotations.

**Definition 6.1** We say that $\Theta$ is a ***suitable metavariable context*** for a set of cf-metavariables $S$ when $S \subseteq |\Theta|$ and $\Theta(\mathsf{M}^{\mathcal{B}}) = \lfloor \mathcal{B} \rfloor$ for all $\mathsf{M}^{\mathcal{B}} \in S$. Similarly, $\Gamma$ is a ***suitable variable context*** for a set of free cf-variables $V$ when $V \subseteq |\Gamma|$ and $\Gamma(\mathsf{a}^A) = \lfloor A \rfloor$ for all $\mathsf{a}^A \in V$. We say that $\Theta; \Gamma$ is a ***suitable context*** for $S$ and $V$ when $\Theta$ is suitable for $S$ an $\Gamma$ for $V$.

As a shorthand, we say that $\Theta; \Gamma$ is *suitable* for a syntactic entity $e$ when it is suitable for $\mathsf{mv}(e)$ and $\mathsf{fv}(e)$. As suitability only depends on the assumption set, it follows from suitability of $\Theta; \Gamma$ for $e$ and $\mathsf{asm}(e') \subseteq \mathsf{asm}(e)$ that $\Theta; \Gamma$ is also suitable for $e'$.

Next, say that a free cf-variable $\mathsf{a}^A$ ***depends*** on a free cf-variable $\mathsf{b}^B$, written $\mathsf{b}^B \prec \mathsf{a}^A$, when $\mathsf{b}^B \in \mathsf{fv}(A)$, and that a set $S$ of free cf-variables is ***closed under dependence*** when $\mathsf{b}^B \prec \mathsf{a}^A \in S$ implies $\mathsf{b}^B \in S$. Every set $S$ of cf-variables is contained in the least closed set, which is $\bigcup \{\!| \mathsf{fv}(\mathsf{a}^A) \mid \mathsf{a}^A \in S |\!\}$. We similarly define dependence for cf-metavariables.

The following lemma shows how to construct suitable contexts.

**Lemma 6.2** *For every finite set of cf-metavariables $S$ there exists a suitable metavariable context $\Theta$, such that $|\Theta|$ is the closure of $S$ with respect to dependence. For every finite set of free cf-variables $V$ there exists a suitable variable context $\Gamma$, such that $|\Gamma|$ is the closure of $V$ with respect to dependence.*

***Proof*** Given a finite set of free cf-variables $S$, the well-founded order $\prec$ on $\bigcup \{\!| \mathsf{fv}(\mathsf{a}^A) \mid \mathsf{a}^A \in S |\!\}$ may be extended to a total one, say $\mathsf{a}_1^{A_1}, \ldots, \mathsf{a}_n^{A_n}$. Now take $\Gamma$ to be the variable context $\mathsf{a}_1^{A_1} : \lfloor A_1 \rfloor, \ldots, \mathsf{a}_n^{A_n} : \lfloor A_n \rfloor$. The argument for metavariables is analogous. □

A totally ordered extension of $\prec$ can be given explicitly, so the preceding proof yields an explicit construction of a suitable contexts. Notice that the construction does not introduce any spurious assumptions, in the sense that for a variable context $\Gamma$ the constructed suitable set $V$ contains only the variables appearing in $\Gamma$ and the annotations of types appearing in $\Gamma$.

**Proposition 6.3** *If $\Theta; \Gamma$ is suitable for a cf-judgement $\mathcal{J}$ then $\Theta; \Gamma \vdash \lfloor \mathcal{J} \rfloor$ is a syntactically valid tt-judgement, and similarly for boundaries.*

***Proof*** A straightforward induction on the structure of the judgement $\mathcal{J}$. □

Next we translate rules, theories, and derivations.

**Proposition 6.4** *A cf-rule and a cf-rule-boundary*

$$\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow j \quad \text{and} \quad \mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow \mathcal{b}$$

*respectively translate to the raw tt-rule and the tt-rule-boundary*

$$\mathsf{M}_1^{\mathcal{B}_1} : \lfloor \mathcal{B}_1 \rfloor, \ldots, \mathsf{M}_n^{\mathcal{B}_n} : \lfloor \mathcal{B}_n \rfloor \Longrightarrow \lfloor j \rfloor$$

*and*

$$\mathsf{M}_1^{\mathcal{B}_1} : \lfloor \mathcal{B}_1 \rfloor, \ldots, \mathsf{M}_n^{\mathcal{B}_n} : \lfloor \mathcal{B}_n \rfloor \Longrightarrow \lfloor \mathcal{b} \rfloor.$$

*A raw-cf theory $T = \langle R_i \rangle_{i \in I}$ over a symbol signature $\Sigma$ is thus translated rule-wise to the raw tt-theory $T_{\mathrm{tt}} = \langle (R_i)_{\mathrm{tt}} \rangle_{i \in I}$ over the same signature.*

***Proof*** The conditions in Definition 4.1 guarantee that $\mathsf{M}_1^{\mathcal{B}_1} : \lfloor \mathcal{B}_1 \rfloor, \ldots, \mathsf{M}_n^{\mathcal{B}_n} : \lfloor \mathcal{B}_n \rfloor$ is a metavariable context and that it is suitable for $\lfloor j \rfloor$ and $\lfloor \mathcal{b} \rfloor$. □

**Theorem 6.5** (Translation from finitary cf- to tt-theories)

1. *The translation of a finitary cf-theory is finitary.*
2. *Suppose $T$ is a finitary cf-theory whose translation $T_{\mathrm{tt}}$ is also finitary. Let $\Theta;\Gamma$ be tt-context such that $\vdash_{T_{\mathrm{tt}}} \Theta\ \mathsf{mctx}$ and $\Theta \vdash_{T_{\mathrm{tt}}} \Gamma\ \mathsf{vctx}$. If $\vdash_T \mathcal{J}$ and $\Theta;\Gamma$ is suitable for $\mathcal{J}$, then $\Theta;\Gamma \vdash_{T_{\mathrm{tt}}} \lfloor \mathcal{J} \rfloor$.*
3. *With $T$, $\Theta;\Gamma$ as in (2), if $\vdash_T \mathcal{B}$ and $\Theta;\Gamma$ is suitable for $\mathcal{B}$ then $\Theta;\Gamma \vdash_{T_{\mathrm{tt}}} \lfloor \mathcal{B} \rfloor$.*

***Proof*** See the proof on Page 89. □

With the theorem in hand, the loose ends are easily tied up.

**Corollary 6.6** *The translation of a standard cf-theory is a standard tt-theory.*

***Proof*** The translation takes symbol rules to symbol rules, and equality rules to equality rules. □

**Corollary 6.7** *If a finitary cf-theory $T$ derives $\vdash_T \mathcal{J}$ and $\mathcal{J}$ has well-typed annotations then there exists a context $\Theta;\Gamma$ which is suitable for $\mathcal{J}$ such that $\vdash_{T_{\mathrm{tt}}} \Theta\ \mathsf{mctx}$ and $\Theta \vdash_{T_{\mathrm{tt}}} \Gamma\ \mathsf{vctx}$.*

***Proof*** We may use the suitable context $\Theta;\Gamma$ with $\Theta$ and $\Gamma$ constructed respectively from $\mathsf{mv}(\mathcal{J})$ and $\mathsf{fv}(\mathcal{J})$ as in Lemma 6.2. □

### 6.2 Translation from tt-Theories to cf-Theories

Transformation from tt-theories to cf-theories requires annotation of variables with typing information, insertion of conversions, and reconstruction of assumption sets. Unlike in the previous section, we cannot directly translate judgements, but must look at derivations in order to tell where conversions should be inserted and what assumption sets used. We begin by defining auxiliary notions that help organize the translation.

Given a cf-expression $e$, let $\llbracket e \rrbracket$ be the ***double erasure*** of $e$, which is like erasure $\lfloor e \rfloor$, except that we also remove annotations: $\llbracket \mathsf{M}^{\mathcal{B}} \rrbracket = \mathsf{M}$ and $\llbracket \mathsf{a}^{A} \rrbracket = \mathsf{a}$. The following definition specifies when an assignment of annotations to variables, which we call a *labeling*, meets the syntactic criteria that makes it eligible for a translation.

**Definition 6.8**

1. Consider a metavariable context
$$\Theta = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_m{:}\mathcal{B}_m].$$
An ***eligible labeling for*** $\Theta$ is a map
$$\theta = \langle \mathsf{M}_1 \mapsto \mathcal{B}'_1, \ldots, \mathsf{M}_m \mapsto \mathcal{B}'_m \rangle$$
which assigns to each $\mathsf{M}_i$ a cf-boundary $\mathcal{B}'_i$ such that $\llbracket \mathcal{B}'_i \rrbracket = \mathcal{B}_i$, and if $\mathsf{M}_j^{\mathcal{B}} \in \mathsf{mv}(\mathcal{B}'_i)$ then $\mathcal{B} = \mathcal{B}'_j$.
2. With $\Theta$ and $\theta$ as above, consider a variable context
$$\Gamma = [\mathsf{a}_1{:}A_1, \ldots, \mathsf{a}_n{:}A_n],$$
over $\Theta$. An ***eligible labeling for*** $\Gamma$ with respect to $\theta$ is a map
$$\gamma = \langle \mathsf{a}_1 \mapsto A'_1, \ldots, \mathsf{a}_n \mapsto A'_n \rangle$$
which assigns to each $\mathsf{a}_i$ a cf-type $A'_i$ such that $\llbracket A'_i \rrbracket = A_i$, if $\mathsf{M}_j^{\mathcal{B}} \in \mathsf{mv}(A_i)$ then $\mathcal{B} = \theta(\mathsf{M}_j)$, and if $\mathsf{a}_k^{A} \in \mathsf{fv}(A_i)$ then $A = \gamma(\mathsf{a}_k)$.

3. A pair $(\theta, \gamma)$ is an ***eligible labeling for*** $\Gamma; \Theta$ when $\theta$ is eligible for $\Theta$ and $\gamma$ is eligible for $\Gamma$ with respect to $\theta$.
4. With $(\theta, \gamma)$ eligible for $\Theta; \Gamma$, an ***eligible cf-judgement*** $\mathcal{J}'$ for a tt-judgement $\mathcal{J}$ over $\Theta; \Gamma$ is one that satisfies $[\![\mathcal{J}']\!] = \mathcal{J}$, if $\mathsf{M}_i^{\mathcal{B}} \in \mathsf{mv}(\mathcal{J}')$ then $\mathcal{B} = \theta(\mathsf{M}_i)$, and if $\mathsf{a}_k^A \in \mathsf{fv}(\mathcal{J}')$ then $A = \gamma(\mathsf{a}_k)$.
5. With $(\theta, \gamma)$ eligible for $\Theta; \Gamma$, an ***eligible cf-boundary*** $\mathcal{B}'$ for a tt-boundary $\mathcal{B}$ over $\Theta; \Gamma$ is one that satisfies $[\![\mathcal{B}']\!] = \mathcal{B}$, if $\mathsf{M}_i^{\mathcal{B}''} \in \mathsf{mv}(\mathcal{B}')$ then $\mathcal{B}'' = \theta(\mathsf{M}_i)$, and if $\mathsf{a}_k^A \in \mathsf{fv}(\mathcal{B}')$ then $A = \gamma(\mathsf{a}_k)$.

We also postulate eligibility requirements for raw rules and theories.

**Definition 6.9** Consider a raw tt-rule

$$R = (\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n \Longrightarrow j).$$

An ***eligible raw cf-rule*** for $R$ is a raw cf-rule

$$R' = (\mathsf{M}_1^{\mathcal{B}'_1}, \ldots, \mathsf{M}_n^{\mathcal{B}'_n} \Longrightarrow j')$$

such that $\theta = \langle \mathsf{M}_1 \mapsto \mathcal{B}'_1, \ldots, \mathsf{M}_n \mapsto \mathcal{B}'_n \rangle$ is eligible for $[\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$, and $j'$ is eligible for $j$ with respect to $\theta$ (and the empty labeling for $[\,]$).

Let $T = \langle R_i \rangle_{i \in I}$ be a raw tt-theory over $\Sigma$. An ***eligible raw cf-theory*** for $T$ is a raw cf-theory $T' = \langle R'_i \rangle_{i \in I}$ over $\Sigma$ such that each $R'_i$ is eligible for $R_i$.

**Theorem 6.10** (Translation of standard tt- to cf-theories)

1. *For any standard tt-theory $T$ there exists a standard cf-theory $T'$ eligible for $T$.*
2. *For any $T$, $T'$ as above, if $\vdash_T \Theta\ \mathsf{mctx}$ then there exists an eligible labeling $\theta$ for $\Theta$ such that $\vdash_{T'} \theta(\mathsf{M})$ for every $\mathsf{M} \in |\Theta|$.*
3. *For any $T$, $T'$, $\Theta$, $\theta$ as above, if $\Theta; [\,] \vdash_T \Gamma\ \mathsf{vctx}$ then there exists an eligible labeling $\gamma$ for $\Gamma$ with respect to $\theta$ such that $\vdash_{T'} \gamma(\mathsf{a})\ \mathsf{type}$ for every $\mathsf{a} \in |\Gamma|$.*
4. *For any $T$, $T'$, $\Theta$, $\theta$, $\Gamma$, $\gamma$ as above, if $\Theta; \Gamma \vdash_T \mathcal{B}$ then there exists an eligible cf-boundary $\mathcal{B}'$ for $\mathcal{B}$ with respect to $\theta$, $\gamma$ such that $\vdash_{T'} \mathcal{B}'$.*
5. *For any $T$, $T'$, $\Theta$, $\theta$, $\Gamma$, $\gamma$, as above, if $\Theta; \Gamma \vdash_T \mathcal{J}$ then there exists an eligible cf-judgement $\mathcal{J}'$ for $\mathcal{J}$ with respect to $\theta$, $\gamma$ such that $\vdash_{T'} \mathcal{J}'$.*

***Proof*** See the proof on Page 93. □

### 6.3 Transporting Meta-theorems Across the Correspondence

In Sect. 5 we proved enough meta-theorems about cf-theories to secure the translations between cf- and tt-theories. We may now take advantage of the translations by transporting meta-theorems about tt-theories to their cf-counterparts. We illustrate the technique by proving the cf-counterpart of Theorem 3.17, which states that judgementally equal derivations act equally on judgements, and by formulating the economic congruence cf-rules.

**Proposition 6.11** *In a standard cf-theory, consider derivable instantiations*

$$I = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto f_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto f_n \rangle \quad \textit{and} \quad J = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto g_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto g_n \rangle$$

*such that $\vdash \mathcal{B}_i$ for each $i = 1, \ldots, n$, as well as*

$$\vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i \equiv g'_i\ \mathsf{by}\ \alpha_i} \qquad \textit{and} \qquad \lfloor g'_i \rfloor = \lfloor g_i \rfloor. \tag{6.1}$$

*If an object cf-judgement $\mathcal{B}\boxed{e}$ has well-typed annotations and is derivable then there is a derivable equality $\mathcal{B}'\boxed{e_I \equiv e_J \text{ by } \beta}$ such that $\beta \subseteq \mathsf{asm}(\mathcal{J}, \vec{f}, \vec{g}, \vec{g}', \vec{\alpha})$, $\lfloor \mathcal{B}' \rfloor = \lfloor I_* \mathcal{B} \rfloor$, $\lfloor e_I \rfloor = \lfloor I_* e \rfloor$ and $\lfloor e_J \rfloor = \lfloor J_* e \rfloor$.*

***Proof*** Let $\Theta; \Gamma$ be a context which is suitable for both (6.1) and $\mathcal{B}\boxed{e}$, and is minimal in the sense that any variable appearing in it also appears in (6.1) or $\mathcal{B}\boxed{e}$. Let $\Xi = \langle \mathsf{M}_1^{\mathcal{B}_1} : \lfloor \mathcal{B}_1 \rfloor, \ldots, \mathsf{M}_n^{\mathcal{B}_n} : \lfloor \mathcal{B}_n \rfloor \rangle$. By Theorem 6.10, erasure yields judgementally equal derivable tt-instaniations $\lfloor I \rfloor$ and $\lfloor J \rfloor$ of $\Xi$ over $\Theta; \Gamma$, and a derivable judgement $\Theta; \Gamma \vdash \lfloor \mathcal{B}\boxed{e} \rfloor$. By Theorem 3.17, the tt-equality

$$\Theta; \Gamma \vdash \lfloor I_* \mathcal{B} \rfloor \boxed{\lfloor I_* e \rfloor \equiv \lfloor J_* e \rfloor}$$

is derivable. We apply the renaming $\mathsf{M}_i^{\mathcal{B}_i} \mapsto \mathsf{M}_i$ and $\mathsf{a}_i^{A_i} \mapsto \mathsf{a}_i$ to it and obtain

$$\Theta; \Gamma \vdash [\![ I_* \mathcal{B} ]\!] \boxed{[\![ I_* e ]\!] \equiv [\![ J_* e ]\!]}.$$

Next, we apply Theorem 6.10 to the above equation with labelings $\theta(\mathsf{M}_i) = \mathcal{B}_i$ and $\gamma(\mathsf{a}_i) = A_i$, which results in a derivable cf-equality

$$\vdash \mathcal{B}' \boxed{e_I \equiv e_J \text{ by } \beta}. \tag{6.2}$$

such that $\lfloor \mathcal{B}' \rfloor = \lfloor I_* \mathcal{B} \rfloor$, $\lfloor e_I \rfloor = \lfloor I_* e \rfloor$ and $\lfloor e_J \rfloor = \lfloor J_* e \rfloor$. Because we required $\Theta; \Gamma$ to be minimal, $\beta$ satisfies the desired constraint. □

The previous proposition gives us a forward-chaining style of congruence rule, because the conclusion is calculated from the premises via the translation theorems. There is also a backward-chaining version in which we proceed from a given (well-formed) cf-equality that we wish to establish.

**Corollary 6.12** *In a standard cf-theory, consider derivable instantiation*

$$I = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto f_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto f_n \rangle \quad \text{and} \quad J = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto g_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto g_n \rangle$$

*such that $\vdash \mathcal{B}_i$ for each $i = 1, \ldots, n$, as well as*

$$\vdash (I_{(i)*} \mathcal{B}_i) \boxed{f_i \equiv g_i' \text{ by } \alpha_i} \quad \text{and} \quad \lfloor g_i' \rfloor = \lfloor g_i \rfloor. \tag{6.3}$$

*Suppose $\vdash \mathcal{B}' \boxed{e_I \equiv e_J \text{ by } \square}$ is derivable, where $\lfloor \mathcal{B}' \rfloor = \lfloor I_* \mathcal{B} \rfloor$, $\lfloor e_I \rfloor = \lfloor I_* e \rfloor$ and $\lfloor e_J \rfloor = \lfloor J_* e \rfloor$. Then there is $\beta \subseteq \mathsf{asm}(\mathcal{J}, \vec{f}, \vec{g}, \vec{g}', \vec{\alpha})$ such that $\vdash \mathcal{B}' \boxed{e_I \equiv e_J \text{ by } \beta}$ is derivable.*

***Proof*** By Proposition 6.11 there is a derivable judgement

$$\vdash \mathcal{B}' \boxed{e_I' \equiv e_J' \text{ by } \beta'}$$

such that $\lfloor \mathcal{B}'' \rfloor = \lfloor I_* \mathcal{B} \rfloor$, $\lfloor e_I' \rfloor = \lfloor I_* e \rfloor$, $\lfloor e_J' \rfloor = \lfloor J_* e \rfloor$, and $\beta$ satisfies that required condition. Apply Theorem 5.17 to rectify the boundary to the given one. □

The method works on other meta-theorems, too. For example, the backward-chaining cf-variant of economic congruence tt-rules (Proposition 3.22) goes as follows.

**Proposition 6.13** *In a standard cf-theory, consider a derivable finitary object rule*

$$\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow \mathcal{C}\boxed{e}$$

*and instantiations of its premises*

$$I = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto f_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto f_n \rangle, \quad \textit{and} \quad J = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto g_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto g_n \rangle.$$

*Suppose the following are derivable:*

1. $\vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i}$ *for each equality boundary* $\mathcal{B}_i$,
2. $\vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i \equiv g_i' \ \mathsf{by}\ \alpha_i}$ *with* $\lfloor g_i' \rfloor = \lfloor g_i \rfloor$ *for each object boundary* $\mathcal{B}_i$.

*Suppose* $\vdash \mathcal{b}'\boxed{e_I \equiv e_J \ \mathsf{by}\ \square}$ *is derivable, where* $\lfloor \mathcal{b}' \rfloor = \lfloor I_* \mathcal{b} \rfloor$, $\lfloor e_I \rfloor = \lfloor I_* e \rfloor$, $\lfloor e_j \rfloor = \lfloor J_* e \rfloor$. *Then there is* $\beta \subseteq \mathsf{asm}(\mathcal{b}\boxed{e}, \vec{f}, \vec{g}, \vec{g'}, \vec{\alpha})$ *such that* $\vdash \mathcal{b}'\boxed{e_I \equiv e_J \ \mathsf{by}\ \beta}$ *is derivable.*

***Proof*** We proceed much as in the proof of Proposition 6.11 and Corollary 6.12, except that we apply Proposition 3.22 on the tt- side. □

## 7 Related and Future Work

Our investigation into a general metatheory for type theory has lead us to present and study two languages. In Sect. 2, we gave a general definition of a broad class of finitary type theories and proved that it satisfies the expected desirable type theoretic meta-theorems. In Sect. 4, we introduced a context-free formulation of type theories and demonstrated that this definition satisfies further meta-theorems, notably strengthening and a context-free inversion principle. Context-free type theories serve as the theoretical foundation of Andromeda 2, as the annotation discipline for variables and metavariables turned out to be better suited for an effectful meta-language [24]. See in particular [24, Chapter 4] for a discussion of the implementation of context-free type theories in Andromeda 2. The generality of finitary type theories has been put to work in [7], where a general equality checking algorithm is shown to be sound for all standard type theories.

Our work was developed concurrently with several other general frameworks for type theory. There are different approaches to the study of formal systems such as logics and type theories, ranging from syntactic [11, 23] to semantic [9, 10, 18, 25] characterisations. To reasonably delimit the scope of this discussion we shall focus on those that (i) are sufficiently expressive to faithfully represent a wide family of dependent type theories, but (ii) are sufficiently restrictive to prove general meta-theorems that are comparable to ours.

### General Dependent Type Theories

The closest relative are general dependent type theories [6], which we proposed together with Lumsdaine. Finitary and general dependent type theories (GDTTs) have more in common than divides them. FTTs can be seen as a bridge from GDTTs to context-free type theories (CFTTs). As context-free type theories in turn are intended as the theoretical underpinning of Andromeda 2, the choice was made to restrict arities of rules and symbols to be finite, which allows for a direct representation as concrete syntax. This restriction is somewhat coincidental, and we expect that it should be possible to generalise much of the treatment of FTTs and possibly CFTTs to arbitrary arities.

The treatment of variables and metavariables in FTTs differs from that of GDTTs in an inessential way: the former uses a locally-nameless discipline and metavariable contexts, while the latter uses shape systems and metavariables as theory extensions. Once again the

difference is motivated by implementation details and the rôle metavariables play in proof assistants.

Finally, the levels of well-formedness of the two formalisms differs slighly. GDTTs places fewer restrictions on the rules of raw type theories, while raw FTTs already satisfies presuppositivity.

We expect that translations between the finitary fragment of GDTTs and FTTs can be defined under mild assumptions, and leave their formal comparison as future work.

### Logical Frameworks

Perhaps the most prominent family of systems for representing logics are logical frameworks [23, 32]. Logical frameworks have spawned a remarkably fruitful line of work [13, 16, 40] and several implementations exist [31, 33]. In concurrent work to the development of GDTTs and FTTs, Uemura [38] and Harper [21] recently proposed frameworks with the purpose of representing type theories.

Both Uemura's LF (*ULF* for short), and Harper's Equational LF (henceforth *EqLF*) extend previous frameworks by the addition of an equality type satisfying reflection to judgemental equality at the framework level, and Uemura includes a substantial development of a general categorical semantics. Harper's Equational LF *almost* forms a standard finitary type theory. In fact, only inessential modifications are needed to put it in standard form, as is confirmed by a formalisation of EqLF in Andromeda 2 [24]. We compare both accounts of type theory to FTTs along several axes. As they are quite similar, we focus on Uemura's variant.

In one way, ULF is more expressive than FTTs. While FTTs allow only one judgement form for types, terms, and their equalities, ULF can also capture theories with other judgement forms, such as the fibrancy judgement of the homotopy type system or two-level type theory [4, 39], or the face formulas of cubical type theory [15]. While it may be possible to reconstruct some type theories expressible in ULF via the use of universes in FTTs, a careful analysis would be required to show that the account is faithful, for instance by showing that it is sound and complete for derivability. Conversely, every standard finitary type theory is expressible in ULF. The translation is straightforward, and we take this as a sign that both ULF and FTTs achieve their goal of giving a "natural" account of type theory.

Finitary type theories on the other hand are not directly expressible in ULF or in EqLF. Frequently, accounts of type theory present rules that are not standard, most often because a symbol does not record all of the metavariables introduced by its premises as arguments. But it is also standard practice to have only one notation for say dependent products which may occur at more than one sort, as is done in [21, 27], or give a general cumulativity rule allowing the silent inclusion of types from one sort into another [26, 38]. One may of course take the view that such presentations are not *really* type theories and should be read with full annotations inserted. It is usually understood that such an annotated presentation can be given, and by including the right set of equations the original calculus can be recovered [22]. Proofs that an unannotated theory is equivalent to a fully annotated one are hard labour [35, Theorem 4.13]. Finitary type theories can thus serve to study the elaboration of such unannotated to a standard FTT or ULF presentation. One such useful general result can already be found in [6], where it is shown that every raw type theory, possibly containing cyclic dependencies between rules, is equivalent to a well-founded one. The assumption of well-founded stratification is hardwired in ULF through the definition of a signature and in EqLF through the inductive construction of a context serving as signature, so that such a theorem could not even be stated in ULF or EqLF. In ongoing research, Petković Komel is

employing finitary type theories to investigate a general elaboration theorem, stating that all finitary type theories can be elaborated to standard ones [29].

It would be useful to prove a general adequacy theorem of Uemura's or Harper's [21] logical framework for finitary type theories. Conversely, the extension of finitary and context-free type theories to other judgement forms in the style of Uemura's LF seems within reach and would allow the expression of exciting new type theories such as those based on cubical sets [3, 12, 15]. Another active domain of current research are modal type theories [8, 34]. Multimodal type theory does not readily fit into our setup or the framework of Uemura [20], and the development of modal finitary type theories is an exciting possibility for further work.

### Context-Free Type Theories

Geuvers et al. [19] investigated the $\Gamma_\infty$ system, a context-free formulation of pure type systems. They prove similar meta-theorems, including translations from and to traditional pure type systems. Pure type systems disallow proof-irrelevant rules such as equality reflection. Consequently, the results of [19] are obtained more straightforwardly and without complications arising from the use of conversion terms and assumption sets. Like the authors of [19], our motivation for avoiding explicit contexts came from implementation considerations. A previous version of Andromeda implemented a form of extensional type theory with assumption sets [5]. The results of [19] have been formalised in the Coq proof assistant. A formalisation of context-free type theories could serve as trusted nucleus of a future version of Andromeda. Generalisations of finitary type theories to more general judgement forms in the style of [38] should be mirrored by the development of the corresponding context-free notions and eventually implemented in Andromeda.

**Acknowledgements** The present work draws its inspiration from our joint work with Peter LeFanu Lumsdaine on general type theories [6]. We thank Peter for spearheading the development of general type theories, which inspired us to implement user-definable dependent type theories in Andromeda 2. We also thank Anja Petković Komel for numerous fruitful discussions, and for pushing through even the most horrid technicalities with us. The theorems about admissibility of substitutions and instantiations are to be considered joint work with Anja. We are grateful to Robert Harper and Matija Pretnar for valuable comments on an earlier version of this material as included in [24]. Finally, we are also grateful to the anonymous reviewers of the Journal of Automated Reasoning for their detailed and helpful feedback.

**Funding** This material is based upon work supported by the Air Force Office of Scientific Research under Award numbers FA9550-14-1-0096 and FA9550-21-1-0024.



## Appendix A: Proofs of Statements

We provide here without further comment the rather technical detailed proofs that were elided in the main text.

### A.1 Proofs of Meta-theorems About Type Theories

This section provides missing proofs from Sect. 3.

**Lemma 3.4** *If a raw type theory derives* $\Theta;\ \Gamma, \mathsf{a}{:}A,\ \Delta \vdash \mathcal{J}$ *and* $\Theta;\ \Gamma \vdash t : A$ *then it derives* $\Theta;\ \Gamma,\ \Delta[t/\mathsf{a}] \vdash \mathcal{J}[t/\mathsf{a}]$.

***Proof*** We proceed by induction on the derivation of the judgement. The induction is mutual with the corresponding statement for boundaries, Lemma 3.5.

*Case* TT- VAR: If the derivation ends with the variable rule for $\mathsf{a}$ then we apply weakening to $\Theta;\ \Gamma \vdash t : A$ to get $\Theta;\ \Gamma,\ \Delta[t/\mathsf{a}] \vdash t : A$. For other variables, we apply the variable rule for the same variable.

*Case* TT- ABSTR: Consider a derivation which ends with an abstraction

$$\frac{\Theta;\ \Gamma, \mathsf{a}{:}A,\ \Delta \vdash B\ \mathsf{type} \qquad \mathsf{b} \notin |\Gamma, \mathsf{a}{:}A,\ \Delta| \qquad \Theta;\ \Gamma, \mathsf{a}{:}A,\ \Delta, \mathsf{b}{:}B \vdash \mathcal{J}[\mathsf{b}/x]}{\Theta;\ \Gamma, \mathsf{a}{:}A,\ \Delta \vdash \{x{:}B\}\ \mathcal{J}}$$

The induction hypotheses for the premises yield

$$\Theta;\ \Gamma,\ \Delta[t/\mathsf{a}] \vdash B[t/\mathsf{a}]\ \mathsf{type} \quad \text{and} \quad \Theta;\ \Gamma,\ \Delta[t/\mathsf{a}],\ \mathsf{b}{:}B[t/\mathsf{a}] \vdash (\mathcal{J}[\mathsf{b}/x])[t/\mathsf{a}].$$

Note that $(\mathcal{J}[\mathsf{b}/x])[t/\mathsf{a}] = (\mathcal{J}[t/\mathsf{a}])[\mathsf{b}/x]$, because $x$ does not occur in $t$, and $\mathsf{a} \neq \mathsf{b}$. Hence abstracting $\mathsf{b}$ in the second premise yields

$$\Theta;\ \Gamma,\ \Delta[t/\mathsf{a}] \vdash \{x{:}B[t/\mathsf{a}]\}\ \mathcal{J}[t/\mathsf{a}],$$

as desired.

*Case* TT- META *and* TT- META- CONGR: We only consider the congruence rules, as the metavariable rule is treated similarly. Consider a derivation which ends with the congruence rule for a metavariable $\mathsf{M}$ whose boundary is $\Theta(\mathsf{M}) = \{\vec{x}{:}\vec{B}\}\ \mathcal{b}$:

$$\frac{\begin{array}{c}\Theta;\ \Gamma, \mathsf{a}{:}A,\ \Delta \vdash s_j : B_j[\vec{s}_{(j)}/\vec{x}_{(j)}] \quad \text{for } j = 1, \ldots, m \\ \Theta;\ \Gamma, \mathsf{a}{:}A,\ \Delta \vdash t_j : B_j[\vec{t}_{(j)}/\vec{x}_{(j)}] \quad \text{for } j = 1, \ldots, m \\ \Theta;\ \Gamma, \mathsf{a}{:}A,\ \Delta \vdash s_j \equiv t_j : B_j[\vec{s}_{(j)}/\vec{x}_{(j)}] \quad \text{for } j = 1, \ldots, m \\ \Theta;\ \Gamma, \mathsf{a}{:}A,\ \Delta \vdash C[\vec{s}/\vec{x}] \equiv C[\vec{t}/\vec{x}] \quad \text{if } \mathcal{b} = \square : C\end{array}}{\Theta;\ \Gamma, \mathsf{a}{:}A,\ \Delta \vdash (\mathcal{b}[\vec{s}/\vec{x}])\boxed{\mathsf{M}(\vec{s}) \equiv \mathsf{M}(\vec{t})}}$$

We apply the induction hypotheses to the premises, and conclude by TT- META- CONGR for $\mathsf{M}$, applied to $\vec{s}[\mathsf{a}/x]$ and $\vec{t}[\mathsf{a}/x]$, taking into account that in general $(e[u/x])[v/\mathsf{a}] = (e[v/\mathsf{a}])[u[v/\mathsf{a}]/x]$.

*Case of a specific rule*: Consider a derivation ending with the application of a raw rule $R = (\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n \Longrightarrow j)$ with $j = \mathcal{b}\boxed{e}$, instantiated by $I = \langle \mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_n \mapsto e_n \rangle$,

$$\frac{\begin{array}{c}\Theta;\ \Gamma, \mathsf{a}{:}A,\ \Delta \vdash (I_{(i)*}\mathcal{B}_i)\boxed{e_i} \quad \text{for } i = 1, \ldots, n \\ \Theta;\ \Gamma, \mathsf{a}{:}A,\ \Delta \vdash I_*\mathcal{b}\end{array}}{\Theta;\ \Gamma, \mathsf{a}{:}A,\ \Delta \vdash I_* j}$$

The induction hypotheses for the premises yield, for $i = 1, \ldots, n$,

$$\Theta;\ \Gamma,\ \Delta[t/\mathsf{a}] \vdash ((I_{(i)*}\mathcal{B}_i)\boxed{e_i})[t/\mathsf{a}],$$

which equals

$$\Theta;\ \Gamma,\ \Delta[t/\mathsf{a}] \vdash (I[t/\mathsf{a}]_{(i)*}\mathcal{B}_i)\boxed{e_i[t/\mathsf{a}]}.$$

By Lemma 3.5, we further obtain $\Theta; \Gamma, \Delta \vdash (I[t/\mathsf{a}])_* \mathcal{b}$. Now apply $R$ instantiated at $I[t/a] = \langle \mathsf{M}_1 \mapsto e_1[t/\mathsf{a}], \ldots, \mathsf{M}_n \mapsto e_n[t/\mathsf{a}] \rangle$ to derive $\Theta; \Gamma, \Delta[t/\mathsf{a}] \vdash I[t/a]_* j$, which equals $\Theta; \Gamma, \Delta[t/\mathsf{a}] \vdash (I_* j)[t/\mathsf{a}]$.

*Case of a congruence rule*: Apply the induction hypotheses to the premises and conclude by the same rule.

*Cases* TT- EqTy- Refl, TT- EqTy- Sym, TT- EqTy- Trans, TT- EqTm- Refl, TT- EqTm- Sym, TT- EqTm- Trans, TT- Conv- Tm, TT- Conv- EqTm: These cases are dispensed with, once again, by straightforward applications of the induction hypotheses. □

**Lemma 3.6** *In a raw type theory the following closure rules are admissible:*

$$\text{TT- Subst}\ \frac{\Theta; \Gamma \vdash \{x{:}A\}\ \mathcal{J} \qquad \Theta; \Gamma \vdash t : A}{\Theta; \Gamma \vdash \mathcal{J}[t/x]} \qquad \text{TT- Bdry- Subst}\ \frac{\Theta; \Gamma \vdash \{x{:}A\}\ \mathcal{B} \qquad \Theta; \Gamma \vdash t : A}{\Theta; \Gamma \vdash \mathcal{B}[t/x]}$$

$$\text{TT- Conv- Abstr}\ \frac{\Theta; \Gamma \vdash \{x{:}A\}\ \mathcal{J} \qquad \Theta; \Gamma \vdash B\ \mathsf{type} \qquad \Theta; \Gamma \vdash A \equiv B}{\Theta; \Gamma \vdash \{x{:}B\}\ \mathcal{J}}$$

***Proof*** Suppose the premises of TT- Subst are derivable. By inversion the first premise is derived by an application of TT- Abstr, therefore for some $\mathsf{a} \notin |\Gamma|$, we can derive $\Theta; \Gamma, \mathsf{a}{:}A \vdash \mathcal{J}[\mathsf{a}/x]$. Lemma 3.4 yields $\Theta; \Gamma \vdash (\mathcal{J}[\mathsf{a}/x])[t/\mathsf{a}]$, which is equal to the conclusion of TT- Subst.

The rule TT- Bdry- Subst follows from Lemma 3.5.

Next, assuming the premises of TT- Conv- Abstr are derivable, its conclusion is derived as

$$\dfrac{\Theta; \Gamma \vdash B\ \mathsf{type} \qquad \dfrac{\dfrac{\Theta; \Gamma \vdash \{x{:}A\}\ \mathcal{J}}{\Theta; \Gamma, \mathsf{a}{:}B \vdash \{x{:}A\}\ \mathcal{J}} \qquad \dfrac{\Theta; \Gamma \vdash A \equiv B}{\Theta; \Gamma \vdash B \equiv A}}{\Theta; \Gamma, \mathsf{a}{:}B \vdash \mathsf{a}{:}A}}{\Theta; \Gamma \vdash \{x{:}B\}\ \mathcal{J}}\ \text{TT- Subst}$$

□

**Lemma 3.7** *If a raw type theory derives*

$$\Theta; \Gamma \vdash s : A, \tag{3.1}$$
$$\Theta; \Gamma \vdash t : A, \tag{3.2}$$
$$\Theta; \Gamma \vdash s \equiv t : A. \tag{3.3}$$
$$\Theta; \Gamma, \mathsf{a}{:}A, \Delta \vdash \mathcal{J}, \tag{3.4}$$
$$\Theta; \Gamma, \Delta[s/\mathsf{a}] \vdash B[s/\mathsf{a}] \equiv B[t/\mathsf{a}] \quad \textit{for all}\ \mathsf{b} \in |\Delta| \textit{with}\ \Delta(\mathsf{b}) = B, \tag{3.5}$$

*then it derives*

1. $\Theta; \Gamma, \Delta[s/\mathsf{a}] \vdash \mathcal{J}[s/\mathsf{a}]$,
2. $\Theta; \Gamma, \Delta[s/\mathsf{a}] \vdash \mathcal{J}[t/\mathsf{a}]$, *and*
3. $\Theta; \Gamma, \Delta[s/\mathsf{a}] \vdash \mathcal{J}[(s \equiv t)/\mathsf{a}]$ *if* $\mathcal{J}$ *is an object judgement.*

***Proof*** We proceed by induction on the derivation of (3.4).

*Case* TT- VAR: For a variable $\mathsf{b} \in |\Gamma|$, (1) and (2) follow by the same variable rule, while (3) follows by reflexivity for $\mathsf{b}$ and the same variable rule.

For the variable $\mathsf{a}$, the desired judgements are precisely the assumptions (3.1), (3.2), and (3.3) weakened to $\Gamma, \Delta[s/\mathsf{a}]$.

For a variable $\mathsf{b} \in |\Delta|$ with $B = \Delta(\mathsf{b})$, the same variable rule derives $\Theta; \Delta[s/\mathsf{a}] \vdash \mathsf{b} : B[s/\mathsf{a}]$ to satisfy (1), while (2) requires an additional conversion along

$$\Theta; \Gamma, \Delta[s/\mathsf{a}] \vdash B[s/\mathsf{a}] \equiv B[t/\mathsf{a}] \tag{A1}$$

which is just (3.5). To show (3), namely $\Theta; \Gamma, \Delta[s/\mathsf{a}] \vdash \mathsf{b} \equiv \mathsf{b} : B[s/\mathsf{a}]$, we use TT- EQTM- REFL and the variable rule.

*Case* TT- ABSTR: Consider a derivation ending with an abstraction

$$\frac{\Theta; \Gamma, \mathsf{a}{:}A, \Delta \vdash B \ \mathsf{type} \qquad \mathsf{b} \notin |\Gamma, \mathsf{a}{:}A, \Delta| \qquad \Theta; \Gamma, \mathsf{a}{:}A, \Delta, \mathsf{b}{:}B \vdash \mathcal{J}[\mathsf{b}/x]}{\Theta; \Gamma, \mathsf{a}{:}A, \Delta \vdash \{x{:}B\}\ \mathcal{J}}$$

The induction hypothesis (1) applied to the first premise yields

$$\Theta; \Gamma, \Delta[s/\mathsf{a}] \vdash B[s/\mathsf{a}] \ \mathsf{type}, \tag{A2}$$

$$\Theta; \Gamma, \Delta[s/\mathsf{a}] \vdash B[s/\mathsf{a}] \equiv B[t/\mathsf{a}]. \tag{A3}$$

Equation (A3) ensures that the extended variable context $\Delta, \mathsf{b}{:}B$ satisfies (3.5), hence we may use the induction hypothesis (1) for the last premise to show

$$\Theta; \Gamma, \Delta[s/\mathsf{a}], \mathsf{b}{:}B[s/\mathsf{a}] \vdash \mathcal{J}[\mathsf{b}/x][s/\mathsf{a}],$$

which equals

$$\Theta; \Gamma, \Delta[s/\mathsf{a}], \mathsf{b}{:}B[s/\mathsf{a}] \vdash \mathcal{J}[s/\mathsf{a}][\mathsf{b}/x]. \tag{A4}$$

We can thus use the abstraction rule with (A2) and (A4) to derive $\Theta; \Gamma, \Delta[s/\mathsf{a}] \vdash \{x{:}B[s/\mathsf{a}]\}\ \mathcal{J}[s/\mathsf{a}]$, as required.

The derivation of $\Theta; \Gamma, \Delta[s/\mathsf{a}] \vdash \{x{:}B[t/\mathsf{a}]\}\ \mathcal{J}[t/\mathsf{a}]$ is more interesting. We first apply induction hypothesis (2) to the last premise and get

$$\Theta; \Gamma, \Delta[s/\mathsf{a}], \mathsf{b}{:}B[s/\mathsf{a}] \vdash \mathcal{J}[\mathsf{b}/x][t/\mathsf{a}].$$

Abstraction now gets us to $\Theta; \Gamma, \Delta[s/\mathsf{a}] \vdash \{x{:}B[s/\mathsf{a}]\}\ \mathcal{J}[t/\mathsf{a}]$, after which we apply TT- -CONV- ABSTR from Lemma 3.6 to replace $B[s/\mathsf{a}]$ with $B[t/\mathsf{a}]$ using (A3).

Lastly, we use the induction hypothesis (3) for the last premise to derive

$$\Theta; \Gamma, \Delta[s/\mathsf{a}], \mathsf{b}{:}B[s/\mathsf{a}] \vdash (\mathcal{J}[\mathsf{b}/x])[(s \equiv t)/\mathsf{a}],$$

which equals

$$\Theta; \Gamma, \Delta[s/\mathsf{a}], \mathsf{b}{:}B[s/\mathsf{a}] \vdash (\mathcal{J}[(s \equiv t)/\mathsf{a}])[\mathsf{b}/x]. \tag{A5}$$

We may thus apply abstraction to (A2) and (A5) to derive

$$\Theta; \Gamma, \Delta[s/\mathsf{a}] \vdash \{x{:}B[s/\mathsf{a}]\}\ \mathcal{J}[(s \equiv t)/\mathsf{a}],$$

as desired.

*Case* TT- META: Suppose (3.4) concludes with the metavariable rule for $\mathsf{M}$, where $\Theta(\mathsf{M}) = \mathcal{B} = (\{x_1{:}A_1\} \cdots \{x_n{:}A_n\}\ \mathcal{b})$:

$$\frac{\Theta;\ \Gamma,\ \mathsf{a}{:}A,\ \Delta \vdash u_i : A_i[\vec{u}_{(i)}/\vec{x}_{(i)}] \quad \text{for } i = 1, \ldots, n \qquad \Theta;\ \Gamma,\ \mathsf{a}{:}A,\ \Delta \vdash \mathcal{b}[\vec{u}/\vec{x}]}{\Theta;\ \Gamma,\ \mathsf{a}{:}A,\ \Delta \vdash ((\mathcal{b}[\vec{u}/\vec{x}])\boxed{\mathsf{M}(\vec{u})})} \tag{A6}$$

Judgements (1) and (2) are derived by the metavariable rule for $\mathsf{M}$, applied to the corresponding induction hypotheses for the premises of (A6). We address (3) in case $\mathcal{b} = (\square : B)$, and leave the simpler case $\mathcal{b} = (\square\ \mathsf{type})$ to the reader. We thus seek a derivation of

$$\Theta;\ \Gamma,\ \Delta[s/\mathsf{a}] \vdash (\mathsf{M}(\vec{u}) : B[\vec{u}/\vec{x}])[(s \equiv t)/\mathsf{a}]$$

which equals

$$\Theta;\ \Gamma,\ \Delta[s/\mathsf{a}] \vdash \mathsf{M}(\vec{u}[s/\mathsf{a}]) \equiv \mathsf{M}(\vec{u}[t/\mathsf{a}]) : B[\vec{u}[s/\mathsf{a}]/\vec{x}].$$

This is just the conclusion of the congruence rule TT- META- CONGR for $\mathsf{M}$, suitably applied so that its term and term equation premises are precisely the induction hypotheses (1,2,3) for the term premises of (A6), and its type equation premise is obtained by application of the induction hypothesis (3) to the last premise of (A6).

*Case* TT- META- CONGR: If (3.4) ends with a congruence rule for an object metavariable $\mathsf{M}$ then both (1) and (2) follow by the same congruence rule, applied to the respective induction hypotheses for the premises.

*Case of a specific rule*: Suppose (3.4) ends with an application of the raw rule $R = (\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n \Longrightarrow j)$ instantiated with $I = \langle \mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_n \mapsto e_n \rangle$:

$$\frac{\Theta;\ \Gamma,\ \mathsf{a}{:}A,\ \Delta \vdash (I_{(i)*}\mathcal{B}_i)\boxed{e_i} \quad \text{for } i = 1, \ldots, n \qquad \Theta;\ \Gamma,\ \mathsf{a}{:}A,\ \Delta \vdash I_*\mathcal{b} \quad \text{where } j = \mathcal{b}\boxed{e}}{\Theta;\ \Gamma,\ \mathsf{a}{:}A,\ \Delta \vdash I_*j} \tag{A7}$$

We would like to derive

$$\Theta;\ \Gamma,\ \Delta[s/\mathsf{a}] \vdash (I_*j)[s/\mathsf{a}], \tag{A8}$$

$$\Theta;\ \Gamma,\ \Delta[s/\mathsf{a}] \vdash (I_*j)[t/\mathsf{a}], \tag{A9}$$

and in case $j$ is an object judgement, also

$$\Theta;\ \Gamma,\ \Delta[s/\mathsf{a}] \vdash (I_*j)[(s \equiv t)/\mathsf{a}]. \tag{A10}$$

We derive (A8) by $(I[s/\mathsf{a}])_*R$ where $I[s/\mathsf{a}] = \langle \mathsf{M}_1 \mapsto e_1[s/\mathsf{a}], \ldots, \mathsf{M}_n \mapsto e_n[s/\mathsf{a}] \rangle$, as its premises are induction hypotheses. Similarly, (A9) is derived by $(I[t/\mathsf{a}])_*R$. We consider (A10) in case $j = (u : B)$ and leave the simpler case $j = (B\ \mathsf{type})$ to the reader. We thus need to derive

$$\Theta;\ \Gamma,\ \Delta[s/\mathsf{a}] \vdash (I_*u)[s/\mathsf{a}] \equiv (I_*u)[t/\mathsf{a}] : (I_*B)[s/\mathsf{a}], \tag{A11}$$

which we do by applying the congruence rule, where $J = I[s/\mathsf{a}]$ and $K = I[t/\mathsf{a}]$,

$$\frac{\begin{array}{ll}\Theta;\ \Gamma,\ \Delta[s/\mathsf{a}] \vdash (J_{(i)*}\mathcal{B}_i)\boxed{e_i[s/\mathsf{a}]} & \text{for } i = 1, \ldots, n \\ \Theta;\ \Gamma,\ \Delta[s/\mathsf{a}] \vdash (K_{(i)*}\mathcal{B}_i)\boxed{e_i[t/\mathsf{a}]} & \text{for } i = 1, \ldots, n \\ \Theta;\ \Gamma,\ \Delta[s/\mathsf{a}] \vdash (J_{(i)*}\mathcal{B}_i)\boxed{e_i[s/\mathsf{a}] \equiv e_i[t/\mathsf{a}]} & \text{for object boundary } \mathcal{B}_i \\ \Theta;\ \Gamma,\ \Delta[s/\mathsf{a}] \vdash J_*B \equiv K_*B & \end{array}}{\Theta;\ \Gamma,\ \Delta[s/\mathsf{a}] \vdash J_*u \equiv K_*u : J_*B}$$

The first three rows of premises are just the induction hypotheses for the first row of premises of (A7), and the last one is (3) for the last premise of (A7).

*Case of a congruence rule*: Both (1) and (2) are derived by applying the induction hypotheses to the premises and using the congruence rule.

*Case* TT- Conv- Tm: Consider a derivation ending with a conversion

$$\frac{\Theta;\, \Gamma, \mathsf{a}{:}A, \Delta \vdash u : B \qquad \Theta;\, \Gamma, \mathsf{a}{:}A, \Delta \vdash B \equiv C}{\Theta;\, \Gamma, \mathsf{a}{:}A, \Delta \vdash u : C}$$

The judgements $\Theta;\, \Gamma, \Delta[s/\mathsf{a}] \vdash u[s/\mathsf{a}] : C[s/\mathsf{a}]$ and $\Theta;\, \Gamma, \Delta[s/\mathsf{a}] \vdash u[t/\mathsf{a}] : C[t/\mathsf{a}]$ immediately follow from the induction hypothesis and conversion. To derive $\Theta;\, \Gamma, \Delta[s/\mathsf{a}] \vdash (u : C)[(s \equiv t)/\mathsf{a}]$, note that the induction hypothesis (3) for the first premise yields

$$\Theta;\, \Gamma, \Delta[s/\mathsf{a}] \vdash u[s/\mathsf{a}] \equiv u[t/\mathsf{a}] : B[s/\mathsf{a}],$$

and (1) applied to the second premise

$$\Theta;\, \Gamma, \Delta[s/\mathsf{a}] \vdash B[s/\mathsf{a}] \equiv C[s/\mathsf{a}].$$

Thus by equality conversion we conclude $\Theta;\, \Gamma, \Delta[s/\mathsf{a}] \vdash u[s/\mathsf{a}] \equiv u[t/\mathsf{a}] : C[s/\mathsf{a}]$.

*Cases* TT- EqTy- Refl, TT- EqTy- Sym, TT- EqTy- Trans, TT- EqTm- Refl, TT- EqTm- Sym, TT- EqTm- Trans, TT- Conv- EqTm: These cases are dispensed with by straightforward applications of the induction hypotheses. □

**Lemma 3.9** *Suppose a raw type theory derives*

$$\Theta;\, \Gamma \vdash s : A, \qquad \Theta;\, \Gamma \vdash t : A, \quad \textit{and} \quad \Theta;\, \Gamma \vdash s \equiv t : A.$$

1. *If it derives*

   $$\Theta;\, \Gamma \vdash \{x{:}A\}\{\vec{y}{:}\vec{B}\}\ C \equiv D \quad \textit{and} \quad \Theta;\, \Gamma \vdash \{x{:}A\}\{\vec{y}{:}\vec{B}\}\ D\ \mathsf{type}$$

   *then it derives* $\Theta;\, \Gamma \vdash \{\vec{y}{:}\vec{B}[s/x]\}\ C[s/x] \equiv D[t/x]$.
2. *If it derives*

   $$\Theta;\, \Gamma \vdash \{x{:}A\}\{\vec{y}{:}\vec{B}\}\ u \equiv v : C \quad \textit{and} \quad \Theta;\, \Gamma \vdash \{x{:}A\}\{\vec{y}{:}\vec{B}\}\ v : C$$

   *then it derives* $\Theta;\, \Gamma \vdash \{\vec{y}{:}\vec{B}[s/x]\}\ u[s/x] \equiv v[t/x] : C[s/x]$.

***Proof*** We spell out the proof of the first claim only. By substituting $s$ for $x$ in the first assumption we obtain

$$\Theta;\, \Gamma \vdash \{\vec{y}{:}\vec{B}[s/x]\}\ C[s/x] \equiv D[s/x],$$

and by applying TT- Subst- EqTy to the second assumption

$$\Theta;\, \Gamma \vdash \{\vec{y}{:}\vec{B}[s/x]\}\ D[s/x] \equiv D[t/x].$$

These two may be combined to give the desired judgement by unpacking the abstraction, applying transitivity, and packing up the abstraction. □

**Lemma 3.10** *Suppose a raw type theory derives, for* $i = 1, \ldots, n$,

$$\Theta;\, \Gamma \vdash s_i : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}]$$
$$\Theta;\, \Gamma \vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}]$$
$$\Theta;\, \Gamma \vdash s_i \equiv t_i : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}].$$

*If it derives an object judgement* $\Theta; \Gamma \vdash \{\vec{x}{:}\vec{A}\}\ \mathcal{B}\boxed{e}$ *then it derives*

$$\Theta; \Gamma \vdash (\mathcal{B}[\vec{s}/\vec{x}])\boxed{e[\vec{s}/\vec{x}] \equiv e[\vec{t}/\vec{x}]}.$$

***Proof*** First, by inversion on the derivation of $\Theta; \Gamma \vdash \{\vec{x}{:}\vec{A}\}\ \mathcal{B}\boxed{e}$ we see that, for $i = 1, \ldots, n$,

$$\Theta; \Gamma \vdash \{\vec{x}_{(i)}{:}\vec{A}_{(i)}\}\ A_i\ \mathsf{type}.$$

Next, we claim that, for all $j = 1, \ldots, i-1$,

$$\Theta; \Gamma \vdash \{x_j{:}A_j[\vec{s}_{(j)}/\vec{x}_{(j)}]\} \cdots \{x_{i-1}{:}A_{i-1}[\vec{s}_{(j)}/\vec{x}_{(j)}]\} \\ A_i[\vec{s}_{(j)}/\vec{x}_{(j)}] \equiv A_i[\vec{t}_{(j)}/\vec{x}_{(j)}].$$

Indeed, when $j = 1$ the statement reduces to reflexivity, while an application of Lemma 3.9 lets us pass from $j$ to $j+1$. When $j = i$ we obtain

$$\Theta; \Gamma \vdash A_i[\vec{s}_{(i)}/\vec{x}_{(i)}] \equiv A_i[\vec{t}_{(i)}/\vec{x}_{(i)}],$$

and this can be used to show by conversion that $\Theta; \Gamma \vdash t_i : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}]$. Now the goal can be derived by repeated applications of Lemma 3.9. □

**Lemma 3.12** *In a raw type theory, let $I$ be a derivable instantiation of $\Xi$ over context $\Theta; \Gamma$. If $\Xi; \Gamma, \Delta \vdash \mathcal{J}$ is derivable then so is $\Theta; \Gamma, I_*\Delta \vdash I_*\mathcal{J}$, and similarly for boundaries.*

***Proof*** We proceed by structural induction on the derivation of $\Xi; \Gamma, \Delta \vdash \mathcal{J}$, only devoting attention to the metavariable and abstraction rules, as all the other cases are straightforward.

*Case* TT- Meta: Consider an application of a metavariable rule for $\mathsf{M}$ with $\Xi(\mathsf{M}) = (\{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \mathcal{b})$ and $I(\mathsf{M}) = \{\vec{x}\}e$:

$$\frac{\Xi; \Gamma, \Delta \vdash t_j : A_j[\vec{t}_{(j)}/\vec{x}_{(j)}] \quad \text{for } j = 1, \ldots, m \qquad \Xi; \Gamma, \Delta \vdash \mathcal{b}[\vec{t}/\vec{x}]}{\Xi; \Gamma, \Delta \vdash (\mathcal{b}[\vec{t}/\vec{x}])\boxed{\mathsf{M}(\vec{t})}}$$

We need to derive

$$\Theta; \Gamma, I_*\Delta \vdash ((I_*\mathcal{b})[I_*\vec{t}/\vec{x}])\boxed{e[I_*\vec{t}/\vec{x}]}. \tag{A12}$$

By induction hypothesis, for each $j = 1, \ldots, m$,

$$\Theta; \Gamma, I_*\Delta \vdash I_*t_j : (I_*A_j)[I_*\vec{t}_{(j)}/\vec{x}_{(j)}],$$

while derivability of $I$ at $\mathsf{M}$ and weakening by $I_*\Delta$ yield

$$\Theta; \Gamma, I_*\Delta \vdash \{\vec{x}{:}I_*\vec{A}\}\ (I_*\mathcal{b})\boxed{e}. \tag{A13}$$

We now derive (A12) by repeatedly using TT- Subst to substitute $I_*t_i$'s for $x_i$'s in (A13).

*Case* TT- Meta- Congr: Consider an application of a metavariable congruence rule for $\mathsf{M}$ with $\Xi(\mathsf{M}) = (\{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \mathcal{b})$ and $I(\mathsf{M}) = \{\vec{x}\}e$:

$$\frac{\begin{array}{ll}\Xi; \Gamma, \Delta \vdash s_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \Xi; \Gamma, \Delta \vdash t_j : A_j[\vec{t}_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \Xi; \Gamma, \Delta \vdash s_j \equiv t_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \Xi; \Gamma, \Delta \vdash C[\vec{s}/\vec{x}] \equiv C[\vec{t}/\vec{x}] & \text{if } \mathcal{b} = (\square : C)\end{array}}{\Xi; \Gamma, \Delta \vdash (\mathcal{b}[\vec{s}/\vec{x}])\boxed{\mathsf{M}(\vec{s}) \equiv \mathsf{M}(\vec{t})}}$$

We need to derive

$$\Theta;\, \Gamma,\, I_*\Delta \vdash ((I_*\mathcal{b})[I_*\vec{s}/x])\boxed{e[I_*\vec{s}/\vec{x}] \equiv e[I_*\vec{t}/\vec{x}]}.$$

Derivability of $I$ yields

$$\Theta;\, \Gamma,\, I_*\Delta \vdash \{\vec{x}{:}I_*\vec{A}\}\ (I_*\mathcal{b})\boxed{e}. \tag{A14}$$

We may apply Lemma 3.10 to (A14) with terms $I_*\vec{s}$ and $I_*\vec{t}$. The preconditions of the lemma are met by the induction hypotheses for the premises.

*Case* TT- ABSTR: Suppose the derivation ends with an abstraction

$$\frac{\Xi;\, \Gamma,\, \Delta \vdash A\ \mathsf{type} \qquad \mathsf{a} \notin |\Gamma,\, \Delta| \qquad \Xi;\, \Gamma,\, \Delta,\, \mathsf{a}{:}A \vdash \mathcal{J}[\mathsf{a}/x]}{\Xi;\, \Gamma,\, \Delta \vdash \{x{:}A\}\ \mathcal{J}}$$

The induction hypotheses for the premises state

$$\Theta;\, \Gamma,\, I_*\Delta \vdash I_*A\ \mathsf{type} \qquad \text{and} \qquad \Theta;\, \Gamma,\, I_*\Delta,\, \mathsf{a}{:}I_*A \vdash I_*(\mathcal{J}[\mathsf{a}/x]).$$

Because $I_*(\mathcal{J}[\mathsf{a}/x]) = (I_*\mathcal{J})[\mathsf{a}/x]$ we may abstract $\mathsf{a}$ to derive

$$\Theta;\, \Gamma,\, I_*\Delta \vdash \{x{:}I_*A\}\ I_*\mathcal{J}. \qquad \square$$

**Lemma 3.14** *In a raw type theory, consider derivable instantiations $I$ and $J$ of $\Xi = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$ over $\Theta;\, \Gamma$ which are judgementally equal. Suppose that $\vdash\ \Xi\ \mathsf{mctx}$ and $\Theta \vdash\ \Gamma\ \mathsf{vctx}$, and that $\Theta;\, \Gamma,\, \Delta \vdash (I_{(i)*}\mathcal{B}_i)\boxed{J(\mathsf{M}_i)}$ is derivable for $i = 1, \ldots, n$, and additionally that, for all $\mathsf{a} \in |\Delta|$ with $\Delta(\mathsf{a}) = A$, so are*

$$\Theta;\, \Gamma,\, I_*\Delta \vdash I_*A\ \mathsf{type},$$
$$\Theta;\, \Gamma,\, I_*\Delta \vdash J_*A\ \mathsf{type},$$
$$\Theta;\, \Gamma,\, I_*\Delta \vdash I_*A \equiv J_*A$$

*If $\Xi;\, \Gamma,\, \Delta \vdash \mathcal{J}$ is derivable then so are*

$$\Theta;\, \Gamma,\, I_*\Delta \vdash I_*\mathcal{J}, \tag{3.6}$$

$$\Theta;\, \Gamma,\, I_*\Delta \vdash J_*\mathcal{J}, \tag{3.7}$$

$$\Theta;\, \Gamma,\, I_*\Delta \vdash (I \equiv J)_*\mathcal{J} \quad \textit{if } \mathcal{J} \textit{ is an object judgement.} \tag{3.8}$$

***Proof*** Note that (3.6) already follows from Theorem 3.13, so we do not bother to reprove it, but we include the statement because we use it repeatedly. We proceed by structural induction on the derivations of $\vdash\ \Xi\ \mathsf{mctx}$ and $\Xi;\, \Gamma,\, \Delta \vdash \mathcal{J}$.

*Case* TT- VAR: Consider a derivation ending with the variable rule

$$\frac{}{\Xi;\, \Gamma,\, \Delta \vdash \mathsf{a}_i : A_i}.$$

We derive (3.7) by the variable rule, and when $\mathsf{a}_i \in |\Delta|$ a subsequent conversion along $\Theta;\, \Gamma,\, I_*\Delta \vdash I_*A_i \equiv J_*A_i$. The judgement (3.8) holds by TT- EQTM- REFL.

*Case* TT- ABSTR: Consider a derivation ending with an abstraction

$$\frac{\Xi;\, \Gamma,\, \Delta \vdash B\ \mathsf{type} \qquad \mathsf{b} \notin |\Gamma,\, \Delta| \qquad \Xi;\, \Gamma,\, \Delta,\, \mathsf{b}{:}B \vdash \mathcal{J}[\mathsf{b}/y]}{\Xi;\, \Gamma,\, \Delta \vdash \{y{:}B\}\ \mathcal{J}}$$

The induction hypothesis for the first premise yields

$$\Theta;\, \Gamma,\, I_*\Delta \vdash I_*B \text{ type}, \tag{A15}$$
$$\Theta;\, \Gamma,\, I_*\Delta \vdash J_*B \text{ type}, \tag{A16}$$
$$\Theta;\, \Gamma,\, I_*\Delta \vdash I_*B \equiv J_*B. \tag{A17}$$

The extended variable context $\Gamma,\, \Delta,\, \mathsf{b}{:}B$ satisfies the preconditions of the induction hypotheses for the second premise, therefore

$$\Theta;\, \Gamma,\, I_*\Delta,\, \mathsf{b}{:}I_*B \vdash (I_*\mathcal{J})[\mathsf{b}/y], \tag{A18}$$

$$\Theta;\, \Gamma,\, I_*\Delta,\, \mathsf{b}{:}I_*B \vdash (J_*\mathcal{J})[\mathsf{b}/y], \tag{A19}$$

$$\Theta;\, \Gamma,\, I_*\Delta,\, \mathsf{b}{:}I_*B \vdash ((I \equiv J)_*\mathcal{J})[\mathsf{b}/y], \tag{A20}$$

where (A20) is present only when $\mathcal{J}$ is an object judgement. Now (3.8) follows by abstraction from (A15) and (A20). To derive (3.7), we first abstract (A19) to get

$$\Theta;\, \Gamma,\, I_*\Delta \vdash \{y{:}I_*B\}\ J_*\mathcal{J}$$

and then apply TT- Conv- Abstr to convert it along (A17) to derive the desired

$$\Theta;\, \Gamma,\, I_*\Delta \vdash \{y{:}J_*B\}\ J_*\mathcal{J}.$$

*Case of a specific rule*: Consider a specific rule

$$R = (\mathsf{N}_1{:}\mathcal{B}'_1, \ldots, \mathsf{N}_m{:}\mathcal{B}'_m \Longrightarrow \mathcal{b}\boxed{e})$$

and an instantiation $K = \langle \mathsf{N}_1 \mapsto g_1, \ldots, \mathsf{N}_m \mapsto g_m \rangle$. Suppose the derivation ends with the instantiation $K_*R$:

$$\frac{\Xi;\, \Gamma,\, \Delta \vdash (K_{(i)*}\mathcal{B}'_i)\boxed{g_i} \quad \text{for } i = 1, \ldots, m \qquad \Xi;\, \Gamma,\, \Delta \vdash K_*\mathcal{b}}{\Xi;\, \Gamma,\, \Delta \vdash K_*(\mathcal{b}\boxed{e})} \tag{A21}$$

We derive (3.7) by $(J_*K)_*R$ where $J_*K = \langle \mathsf{N}_1 \mapsto J_*g_1, \ldots, \mathsf{N}_m \mapsto J_*g_m \rangle$. The resulting premises for $i = 1, \ldots, m$ are precisely the induction hypotheses (3.7) for the premises of (A21). The last premise, $\Theta;\, \Gamma,\, I_*\Delta \vdash (J_*K)_*\mathcal{b}$, follows by case analysis of $\mathcal{b}$ and the same induction hypothesis (3.7). To establish (3.8), we must derive

$$\Theta;\, \Gamma,\, I_*\Delta \vdash ((I_*K)_*\mathcal{b})\boxed{(I_*K)_*e \equiv (J_*K)_*e}.$$

We do so by an application of the congruence rule associated with $R$, instantiated with $I_*K$ and $J_*K$. The resulting closure rule has four sets of premises, all of which are derivable:

- both copies of premises of $R$ are derivable because they are the induction hypotheses (3.6) and (3.7) for the premises of (A21),
- the additional equational premises are derivable because they are the induction hypotheses (3.8) for the premises of (A21).

*Case of a congruence rule*: Similar to the case of a specific rule. Given a congruence rule with instantiations $L$ and $K$, (3.7) follows from the same congruence rule with instantiations $J_*L$ and $J_*K$. The premises hold by induction hypothesis (3.7).

*Case* TT- META: Consider a derivation ending with an application of the metavariable rule for $\mathsf{M}_i$, where $\vec{x} = (x_1, \ldots, x_m)$, $\vec{t} = (t_1, \ldots, t_m)$, $J(\mathsf{M}_i) = \{\vec{x}\}e$, and $\mathcal{B}_i = \{\vec{x}{:}\vec{A}\}\ \mathcal{b}$,

$$\frac{\Xi;\ \Gamma,\ \Delta \vdash t_j : A_j[\vec{t}_{(j)}/\vec{x}_{(j)}] \quad \text{for } j = 1, \ldots, m \qquad \Xi;\ \Gamma,\ \Delta \vdash \mathcal{b}[\vec{t}/\vec{x}]}{\Xi;\ \Gamma,\ \Delta \vdash (\mathcal{b}[\vec{t}/\vec{x}])\boxed{\mathsf{M}_i(\vec{t})}} \tag{A22}$$

Because $J$ is derivable we know that $\Theta;\ \Gamma \vdash \{\vec{x}{:}J_*\vec{A}\}\ (J_*\mathcal{b})\boxed{e}$. For (3.7), we derive

$$\Theta;\ \Gamma,\ I_*\Delta \vdash ((J_*\mathcal{b})[J_*\vec{t}/\vec{x}])\boxed{e[J_*\vec{t}/\vec{x}]}$$

by substituting $J_*\vec{t}$ for $\vec{x}$ by repeated applications of TT- SUBST, which generate premises, for $j = 1, \ldots, m$,

$$\Theta;\ \Gamma,\ I_*\Delta \vdash J_*t_j : (J_*A_j)[J_*\vec{t}_{(j)}/\vec{x}_{(j)}].$$

These are precisely the induction hypotheses for the premises of (A22). It remains to show (3.8). Writing $I(\mathsf{M}_i)$ as $\{x\}e'$, we must establish

$$\Theta;\ \Gamma,\ I_*\Delta \vdash ((I_*\mathcal{b})[I_*\vec{t}/\vec{x}])\boxed{e'[I_*\vec{t}/\vec{x}] \equiv e[J_*\vec{t}/\vec{x}]}.$$

Because $I$ and $J$ are judgementally equal, we know that

$$\Theta;\ \Gamma \vdash \{\vec{x} : I_*\vec{A}\}\ (I_*\mathcal{b})\boxed{e' \equiv e}.$$

By substituting $I_*\vec{t}$ for $\vec{x}$ by repeated use of TT- SUBST, we derive

$$\Theta;\ \Gamma,\ I_*\Delta \vdash ((I_*\mathcal{b})[I_*\vec{t}/\vec{x}])\boxed{e'[I_*\vec{t}/\vec{x}] \equiv e[I_*\vec{t}/\vec{x}]}, \tag{A23}$$

where the substitutions generate obligations, for $j = 1, \ldots, m$,

$$\Theta;\ \Gamma,\ I_*\Delta \vdash I_*t_j : (I_*A_j)[I_*\vec{t}_{(j)}/\vec{x}_{(j)}].$$

These are precisely the induction hypotheses for the term premises of (A22). By transitivity it suffices to derive

$$\Theta;\ \Gamma,\ I_*\Delta \vdash ((I_*\mathcal{b})[I_*\vec{t}/\vec{x}])\boxed{e[I_*\vec{t}/\vec{x}] \equiv e[J_*\vec{t}/\vec{x}]}. \tag{A24}$$

The induction hypotheses for the premises of (A22) for $j = 1, \ldots, m$ are

$$\Theta;\ \Gamma,\ I_*\Delta \vdash I_*t_j : (I_*A_j)[I_*\vec{t}_{(j)}/\vec{x}_{(j)}] \tag{A25}$$

$$\Theta;\ \Gamma,\ I_*\Delta \vdash J_*t_j : (J_*A_j)[J_*\vec{t}_{(j)}/\vec{x}_{(j)}] \tag{A26}$$

$$\Theta;\ \Gamma,\ I_*\Delta \vdash I_*t_j \equiv J_*t_j : (I_*A_j)[I_*\vec{t}_{(j)}/\vec{x}_{(j)}]. \tag{A27}$$

We would like to apply Lemma 3.10 to these judgements to derive (A24), but the type of the terms $J_*t_j$ in (A26) does not match the type of the corresponding terms $I_*t_j$. We rectify the situation by successively deriving the equality of the types involved and converting, as follows.

By assumption $\vdash \Xi\ \mathsf{mctx}$ holds and hence $\Xi_{(i)};\ [\,] \vdash \{x_1{:}A_1\} \cdots \{x_{j-1}{:}A_{j-1}\}\ A_j\ \mathsf{type}$ for $j = 1, \ldots, m$. Note that the preceding judgement is derivable in a smaller metavariable context, and we can thus appeal to the induction hypothesis to derive

$$\Theta;\ \Gamma,\ I_*\Delta \vdash \{x_1{:}I_*A_1\} \cdots \{x_{j-1}{:}I_*A_{j-1}\}\ I_*A_j \equiv J_*A_j.$$

We apply Lemma 3.10 together with (A25,A26,A27) to obtain

$$\Theta;\ \Gamma,\ I_*\Delta \vdash (I_*A_j)[I_*\vec{t}_{(j)}/\vec{x}_{(j)}] \equiv (J_*A_j)[J_*\vec{t}_{(j)}/\vec{x}_{(j)}].$$

We now appeal to TT- Conv- Tm to derive

$$\Theta; \Gamma, I_*\Delta \vdash J_*t_j : (I_*A_j)[I_*\vec{t}_{(j)}/\vec{x}_{(j)}]. \tag{A28}$$

Finally we derive (A24) by applying Lemma 3.10 to (A25,A28,A27) and to the judgement $\Theta; \Gamma \vdash \{\vec{x}{:}I_*\vec{A}\}(I_*\mathcal{b})\boxed{e}$, which equals $\Theta; \Gamma \vdash (I_*\mathcal{B}_i)\boxed{J(\mathsf{M}_i)}$ and so is derivable by assumption.

*Case* TT- Meta- Congr: Consider a derivation ending with an application of the congruence rule for $\mathsf{M}_i$, where $\vec{x} = (x_1, \ldots, x_m)$, $\vec{s} = (s_1, \ldots, s_m)$, $\vec{t} = (t_1, \ldots, t_m)$, $J(\mathsf{M}_i) = \{\vec{x}\}e$, and $\mathcal{B}_i = \{\vec{x}{:}\vec{A}\}\ \mathcal{b}$,

$$\frac{\begin{array}{ll}\Xi; \Gamma, \Delta \vdash s_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \Xi; \Gamma, \Delta \vdash t_j : A_j[\vec{t}_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \Xi; \Gamma, \Delta \vdash s_j \equiv t_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \Xi; \Gamma, \Delta \vdash C[\vec{s}/\vec{x}] \equiv C[\vec{t}/\vec{x}] & \text{if } \mathcal{b} = (\square : C)\end{array}}{\Xi; \Gamma, \Delta \vdash (\mathcal{b}[\vec{s}/\vec{x}])\boxed{\mathsf{M}_i(\vec{s}) \equiv \mathsf{M}_i(\vec{t})}} \tag{A29}$$

Because $J$ is derivable we know that $\Theta; \Gamma \vdash \{\vec{x}{:}J_*\vec{A}\}\ (J_*\mathcal{b})\boxed{e}$, therefore by weakening also

$$\Theta; \Gamma, I_*\Delta \vdash \{\vec{x}{:}J_*\vec{A}\}\ (J_*\mathcal{b})\boxed{e}.$$

The desired judgement

$$\Theta; \Gamma, I_*\Delta \vdash ((J_*\mathcal{b})[J_*\vec{s}/\vec{x}])\boxed{e[J_*\vec{s}/\vec{x}] \equiv e[J_*\vec{t}/\vec{x}]}$$

may be derived by repeated applications of TT- Subst- EqTm, provided that, for $j = 1, \ldots, m$,

$$\begin{aligned}&\Theta; \Gamma, I_*\Delta \vdash J_*s_j : (J_*A_j)[J_*\vec{s}_{(j)}/\vec{x}_{(j)}], \\ &\Theta; \Gamma, I_*\Delta \vdash J_*t_j : (J_*A_j)[J_*\vec{t}_{(j)}/\vec{x}_{(j)}], \\ &\Theta; \Gamma, I_*\Delta \vdash J_*s_j \equiv J_*t_j : (J_*A_j)[J_*\vec{s}_{(j)}/\vec{x}_{(j)}].\end{aligned}$$

These are precisely induction hypotheses for (A29).

*Cases* TT- EqTy- Refl, TT- EqTy- Sym, TT- EqTy- Trans, TT- EqTm- Refl, TT- EqTm- Sym, TT- EqTm- Trans, TT- Conv- Tm, *and* TT- Conv- EqTm : The remaining cases are all equality rules. Each is established by an appeal to the induction hypotheses for the premises, followed by an application of the same rule. □

**Lemma 3.15** *In a raw type theory, consider* $\Xi = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$ *such that* $\vdash \Xi\ \mathsf{mctx}$, *and derivable instantiations*

$$I = \langle \mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_n \mapsto e_n \rangle \quad \text{and} \quad J = \langle \mathsf{M}_1 \mapsto f_1, \ldots, \mathsf{M}_n \mapsto f_n \rangle$$

*of* $\Xi$ *over* $\Theta; \Gamma$ *which are judgementally equal. Suppose further that* $\Theta \vdash \Gamma\ \mathsf{vctx}$ *and* $\Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i}$ *for* $i = 1, \ldots, n$. *If* $\Theta \vdash (\Gamma, \Delta)\ \mathsf{vctx}$, *then for all* $\mathsf{a} \in |\Delta|$ *with* $\Delta(\mathsf{a}) = A$:

$$\begin{aligned}&\Theta; \Gamma, I_*\Delta \vdash I_*A\ \mathsf{type}, \\ &\Theta; \Gamma, I_*\Delta \vdash J_*A\ \mathsf{type}, \\ &\Theta; \Gamma, I_*\Delta \vdash I_*A \equiv J_*A.\end{aligned}$$

***Proof*** We proceed by induction on the length of $\Delta$. The base case is trivial. For the induction step, suppose $\Theta \vdash (\Gamma, \Delta, \mathsf{b}{:}B)\ \mathsf{vctx}$. For $\mathsf{a} \in |\Delta|$ we apply the induction hypothesis to $\Delta$ and weaken by $\mathsf{b}{:}I_*B$. To deal with $\mathsf{b}$, we apply Lemma 3.14 to $\Theta; \Gamma, \Delta \vdash B\ \mathsf{type}$, which holds by inversion, and weaken by $\mathsf{b}{:}I_*B$ to derive the desired

$$\begin{aligned}
&\Theta; \Gamma, I_*\Delta, \mathsf{b}{:}I_*B \vdash I_*B\ \mathsf{type},\\
&\Theta; \Gamma, I_*\Delta, \mathsf{b}{:}I_*B \vdash J_*B\ \mathsf{type},\\
&\Theta; \Gamma, I_*\Delta, \mathsf{b}{:}I_*B \vdash I_*B \equiv J_*B.
\end{aligned}$$

□

**Lemma 3.16** *In a raw type theory, consider* $\Xi = [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n]$ *such that* $\vdash \Xi\ \mathsf{mctx}$*, and derivable instantiations*

$$I = \langle \mathsf{M}_1 \mapsto e_1, \ldots, \mathsf{M}_n \mapsto e_n \rangle \quad \text{and} \quad J = \langle \mathsf{M}_1 \mapsto f_1, \ldots, \mathsf{M}_n \mapsto f_n \rangle$$

*of* $\Xi$ *over* $\Theta; \Gamma$ *which are judgementally equal. Suppose that* $\Theta \vdash \Gamma\ \mathsf{vctx}$*. Then* $\Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i}$ *is derivable for* $i = 1, \ldots, n$*.*

***Proof*** We proceed by induction on $n$. The base case is trivial. To prove the induction step for $n > 0$, suppose the statement holds for $\Xi_{(n)}$, $I_{(n)}$ and $J_{(n)}$, and that $\mathcal{B}_n = \{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \mathcal{b}$. By inversion on $\vdash \Xi\ \mathsf{mctx}$ and weakening we derive $\Xi_{(n)}; \Gamma \vdash \mathcal{B}_n$. Then by inverting the abstractions of $\mathcal{B}_n$ we obtain variables $\vec{a} = (\mathsf{a}_1, \ldots, \mathsf{a}_m)$ such that, with $A'_i = A_i[\vec{a}_{(i)}/\vec{x}_{(i)}]$ and $\Delta = [\mathsf{a}_1{:}A'_1, \ldots, \mathsf{a}_m{:}A'_m]$,

$$\Xi_{(n)} \vdash (\Gamma, \Delta)\ \mathsf{vctx}, \qquad \text{and} \qquad \Xi_{(n)}; \Gamma, \Delta \vdash \mathcal{b}[\vec{a}/\vec{x}].$$

We apply Lemma 3.15 to $\Xi_{(n)}$, $I_{(n)}$, $J_{(n)}$, and $\Delta$ to derive, for $i = 1, \ldots, m$,

$$\begin{aligned}
&\Theta; \Gamma, I_*\Delta \vdash I_*A'_i\ \mathsf{type},\\
&\Theta; \Gamma, I_*\Delta \vdash J_*A'_i\ \mathsf{type},\\
&\Theta; \Gamma, I_*\Delta \vdash I_*A'_i \equiv J_*A'_i,\\
&\Theta; \Gamma, I_*\Delta \vdash \mathsf{a}_i : J_*A'_i. \qquad \text{(A30)}
\end{aligned}$$

where (A30) follows by conversion from the judgement above it. Next, we use (A30) to substitute $\mathsf{a}_i$ for $x_i$ in $\Theta; \Gamma, I_*\Delta \vdash \{\vec{x}{:}J_*\vec{A}\}\ (J_*\mathcal{b})\boxed{f_n}$, which results in

$$\Theta; \Gamma, I_*\Delta \vdash ((J_*\mathcal{b})[\vec{a}/\vec{x}])\boxed{f_n[\vec{a}/\vec{x}]}. \qquad \text{(A31)}$$

If we can reduce (A31) to

$$\Theta; \Gamma, I_*\Delta \vdash ((I_*\mathcal{b})[\vec{a}/\vec{x}])\boxed{f_n[\vec{a}/\vec{x}]}, \qquad \text{(A32)}$$

we will be able to derive the desired judgement

$$\Theta; \Gamma, I_*\Delta \vdash \{\vec{x}{:}I_*\vec{A}\}\ (I_*\mathcal{b})\boxed{f_n}$$

by abstracting $\mathsf{a}_1, \ldots, \mathsf{a}_n$ in (A32). There are four cases, depending on what $\mathcal{b}$ is.

*Case* $\mathcal{b} = (\square\ \mathsf{type})$: (A31) and (A32) are the same.

*Case* $\mathcal{b} = (\square : B)$: We convert (A31) along

$$\Theta; \Gamma, I_*\Delta \vdash (J_*B)[\vec{a}/\vec{x}] \equiv (I_*B)[\vec{a}/\vec{x}],$$

which holds by Lemma 3.14 applied to $\Xi_{(n)}; \Gamma, \Delta \vdash B[\vec{a}/\vec{x}]\ \mathsf{type}$ with $\Xi_{(n)}$, $I_{(n)}$, and $J_{(n)}$.

*Case* $\mathcal{b} = (B \equiv C \text{ by } \square)$: Here (A31) and (A32) are respectively

$$\Theta; \Gamma, I_*\Delta \vdash J_*B \equiv J_*C \quad \text{and} \quad \Theta; \Gamma, I_*\Delta \vdash I_*B \equiv I_*C.$$

The latter follows from the former if we can also derive

$$\Theta; \Gamma, I_*\Delta \vdash I_*B \equiv J_*B, \quad \text{and} \quad \Theta; \Gamma, I_*\Delta \vdash I_*C \equiv J_*C. \tag{A33}$$

We invert $\Xi_{(n)}; \Gamma, \Delta \vdash B \equiv C \text{ by } \square$ to derive

$$\Xi_{(n)}; \Gamma, \Delta \vdash B \text{ type} \quad \text{and} \quad \Xi_{(n)}; \Gamma, \Delta \vdash C \text{ type}. \tag{A34}$$

When we apply Lemma 3.14 to (A34) it gives us (A33).

*Case* $\mathcal{b} = (s \equiv t : B \text{ by } \square)$: Here (A31) and (A32) are respecetively

$$\Theta; \Gamma, I_*\Delta \vdash J_*s \equiv J_*t : J_*B \quad \text{and} \quad \Theta; \Gamma, I_*\Delta \vdash I_*s \equiv I_*t : I_*B,$$

The latter follows from the former if we can also derive

$$\begin{aligned} &\Theta; \Gamma, I_*\Delta \vdash I_*B \equiv J_*B, \\ &\Theta; \Gamma, I_*\Delta \vdash I_*s \equiv J_*s : I_*B, \\ &\Theta; \Gamma, I_*\Delta \vdash I_*t \equiv J_*t : I_*B. \end{aligned} \tag{A35}$$

We invert $\Xi_{(n)}; \Gamma, \Delta \vdash s \equiv t : B \text{ by } \square$ to derive

$$\Xi_{(n)}; \Gamma, \Delta \vdash B \text{ type}, \quad \Xi_{(n)}; \Gamma, \Delta \vdash s : B \quad \text{and} \quad \Xi_{(n)}; \Gamma, \Delta \vdash t : B. \tag{A36}$$

When we apply Lemma 3.14 to (A36) it gives us (A35). □

**Theorem 3.18** (Presuppositivity) *If a raw type theory derives* $\vdash \Theta$ mctx, $\Theta \vdash \Gamma$ vctx, *and* $\Theta; \Gamma \vdash \mathcal{B}\boxed{e}$ *then it derives* $\Theta; \Gamma \vdash \mathcal{B}$.

***Proof*** We proceed by induction on the derivation of $\Theta; \Gamma \vdash \mathcal{B}\boxed{e}$.

*Case* TT- Var: By Proposition 3.3.

*Case* TT- Meta: The presupposition $\Theta; \Gamma \vdash \mathcal{b}[\vec{t}/\vec{x}]$ is available as premise.

*Case* TT- Meta- Congr: Consider a derivation ending with an application of the congruence rule for $\mathsf{M}$ whose boundary is $\Theta(\mathsf{M}) = (\{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \mathcal{b})$:

$$\frac{\begin{aligned} &\Theta; \Gamma \vdash s_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}] && \text{for } j = 1, \ldots, m \\ &\Theta; \Gamma \vdash t_j : A_j[\vec{t}_{(j)}/\vec{x}_{(j)}] && \text{for } j = 1, \ldots, m \\ &\Theta; \Gamma \vdash s_j \equiv t_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}] && \text{for } j = 1, \ldots, m \\ &\Theta; \Gamma \vdash C[\vec{s}/\vec{x}] \equiv C[\vec{t}/\vec{x}] && \text{if } \mathcal{b} = (\square : C) \end{aligned}}{\Theta; \Gamma \vdash (\mathcal{b}[\vec{s}/\vec{x}])\boxed{\mathsf{M}(\vec{s}) \equiv \mathsf{M}(\vec{t})}}$$

If $\mathcal{b} = (\square \text{ type})$, the presupposition $\Theta; \Gamma \vdash \mathsf{M}(\vec{s}) \equiv \mathsf{M}(\vec{t}) \text{ by } \square$ follows directly by TT- Bdry- EqTy and two uses of TT- Meta. If $\mathcal{b} = (\square : C)$, the presuppositions of $\vdash \mathsf{M}(\vec{s}) \equiv \mathsf{M}(\vec{t}) : C[\vec{s}/\vec{x}] \text{ by } \square$ follow by TT- Bdry- EqTm:

1. $\Theta; \Gamma \vdash C[\vec{s}/\vec{x}]$ type holds by substitution of $\vec{s}$ for $\vec{x}$ in $\Theta; \Gamma \vdash \{\vec{x}{:}\vec{A}\}\ C$ type much like in the previous case,
2. $\Theta; \Gamma \vdash \mathsf{M}(\vec{s}) : C[\vec{s}/\vec{x}]$ holds by TT- Meta,
3. $\Theta; \Gamma \vdash \mathsf{M}(\vec{t}) : C[\vec{s}/\vec{x}]$ is derived from $\Theta; \Gamma \vdash \mathsf{M}(\vec{t}) : C[\vec{t}/\vec{x}]$ by conversion along $\Theta; \Gamma \vdash C[\vec{t}/\vec{x}] \equiv C[\vec{s}/\vec{x}]$, which holds by the last premise.

When applying TT- Meta above, the premise $\Theta; \Gamma \vdash \mathcal{b}[\vec{s}/\vec{x}]$ is required, and likewise for $\vec{t}$. We may derive it by applying Proposition 3.3 to $\vdash \Theta\ \mathsf{mctx}$ and substituting $\vec{s}$ for $\vec{x}$ with the help of TT- Subst, and analogously for $\vec{t}$.

*Case* TT- Abstr: Consider an abstraction

$$\frac{\Theta; \Gamma \vdash A\ \mathsf{type} \qquad \mathsf{a} \notin |\Gamma| \qquad \Theta; \Gamma, \mathsf{a}{:}A \vdash \mathcal{J}[\mathsf{a}/x]}{\Theta; \Gamma \vdash \{x{:}A\}\ \mathcal{J}}$$

By induction hypothesis on the last premise, we obtain $\Theta; \Gamma, \mathsf{a}{:}A \vdash \mathcal{B}[\mathsf{a}/x]$ after which we apply TT- Bdry- Abstr.

*Case of a specific rule*: The presupposition is available as premise.

*Case of a congruence rule*: Consider a congruence rule associated with an object rule $R$ and instantiated with $I$ and $J$, as in Definition 2.17.

If $R$ concludes with $\vdash A\ \mathsf{type}$, the presuppositions are $\Theta; \Gamma \vdash I_* A\ \mathsf{type}$ and $\Theta; \Gamma \vdash J_* A\ \mathsf{type}$, which are derivable by $I_* R$ and $J_* R$, respectively.

If $R$ concludes with $\vdash t : A$, the presuppositions are $\Theta; \Gamma \vdash I_* A\ \mathsf{type}$, $\Theta; \Gamma \vdash I_* t : I_* A$, and $\Theta; \Gamma \vdash J_* t : I_* A$. We derive the first one by applying the induction hypothesis to the premise $\Theta; \Gamma \vdash I_* B \equiv J_* B$, the second one by $I_* R$, and the third one by converting the second one along the aforementioned premise.

*Cases* TT- EqTy- Refl, TT- EqTy- Sym, TT- EqTy- Trans, TT- EqTm- Refl, TT- EqTm- Sym, TT- EqTm- Trans: These are all dispensed with by straightforward appeals to the induction hypotheses.

*Case* TT- Conv- Tm: Consider a term conversion

$$\frac{\Theta; \Gamma \vdash t : A \qquad \Theta; \Gamma \vdash A \equiv B}{\Theta; \Gamma \vdash t : B}$$

Then $\Theta; \Gamma \vdash B\ \mathsf{type}$ holds by the induction hypothesis for the second premise.

*Case* TT- Conv- EqTm: Consider a term equality conversion

$$\frac{\Theta; \Gamma \vdash s \equiv t : A \qquad \Theta; \Gamma \vdash A \equiv B}{\Theta; \Gamma \vdash s \equiv t : B}$$

As in the previous case, the induction hypothesis for the second premise provides $\Theta; \Gamma \vdash B\ \mathsf{type}$. The induction hypothesis for the first premise yields

$$\Theta; \Gamma \vdash s : A \qquad \text{and} \qquad \Theta; \Gamma \vdash t : A$$

We may convert these to $\Theta; \Gamma \vdash s : B$ and $\Theta; \Gamma \vdash t : B$ using the second premise. □

**Proposition 3.20** (Economic version of Definition 2.19) *If a raw type theory derives* $\vdash \Theta\ \mathsf{mctx}$ *and* $\Theta \vdash \Gamma\ \mathsf{vctx}$ *with* $\Theta(\mathsf{M}) = (\{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \mathcal{b})$*, the following closure rules are admissible:*

TT- Meta- Eco

$$\frac{\Theta; \Gamma \vdash t_j : A_j[\vec{t}_{(j)}/\vec{x}_{(j)}] \ \ \textit{for } j = 1, \ldots, m}{\Theta; \Gamma \vdash (\mathcal{b}[\vec{t}/\vec{x}])\boxed{\mathsf{M}(\vec{t})}}$$

TT- Meta- Congr- Eco

$$\frac{\Theta; \Gamma \vdash s_j \equiv t_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}] \ \ \textit{for } j = 1, \ldots, m}{\Theta; \Gamma \vdash (\mathcal{b}[\vec{s}/\vec{x}])\boxed{\mathsf{M}_k(\vec{s}) \equiv \mathsf{M}_k(\vec{t})}}$$

***Proof*** To prove admissibility of TT- Meta- Eco, note that by Proposition 3.3 we have $\Theta; \Gamma \vdash \{\vec{x}{:}\vec{A}\}\ \mathcal{b}$, so we may derive $\Theta; \Gamma \vdash \mathcal{b}[\vec{t}/\vec{x}]$ by substituting $\vec{t}$ for $\vec{x}$ by repeated applications of TT- Bdry- Subst to the premises of TT- Meta- Eco. We can now apply TT- Meta.

Next, we address admissibility of TT- Meta- Congr- Eco by deriving its conclusion with the aid of TT- Meta- Congr. For this purpose we need to derive

$$\begin{aligned} &\Theta; \Gamma \vdash s_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}] && \text{for } j = 1, \ldots, m \\ &\Theta; \Gamma \vdash t_j : A_j[\vec{t}_{(j)}/\vec{x}_{(j)}] && \text{for } j = 1, \ldots, m \\ &\Theta; \Gamma \vdash C[\vec{s}/\vec{x}] \equiv C[\vec{t}/\vec{x}] && \text{if } \mathcal{b} = (\square : C) \end{aligned}$$

The first group follows by Theorem 3.18. The second is established by induction on $j$: by Proposition 3.3, $\Theta \vdash \{\vec{x}{:}\vec{A}\}\ \mathcal{b}$ holds, and thus $\Theta \vdash \{\vec{x}_{(j)}{:}\vec{A}_{(j)}\}\ A_j$ type by inversion of TT- Bdry- Abstr. By applying Lemma 3.10, we obtain $\Theta; \Gamma \vdash A_j[\vec{s}_{(j)}/\vec{x}_{(j)}] \equiv A_j[\vec{t}_{(j)}/\vec{x}_{(j)}]$ and we can convert $\Theta; \Gamma \vdash t_j : A_j[\vec{s}_{(j)}/\vec{x}_{(j)}]$ which holds again by Theorem 3.18. Finally, the last premise holds again by Lemma 3.10, this time applied to $\Theta \vdash \{\vec{x}{:}\vec{A}\}\ C$ type. □

**Lemma 3.21** *In a raw type theory, suppose $\Xi; \Gamma \vdash \mathcal{B}$, and consider judgementally equal derivable instantiations $I, J$ of $\Xi$ over $\Theta; \Gamma$. If $\Theta; \Gamma \vdash (I_*\mathcal{B})\boxed{e}$ is derivable then so is $\Theta; \Gamma \vdash (J_*\mathcal{B})\boxed{e}$.*

***Proof*** We proceed by induction on the derivation of $\Xi; \Gamma \vdash \mathcal{B}$.

*Case* TT- Bdry- Ty: We have $\mathcal{B} = (\square$ type$)$, the statement is trivial.

*Case* TT- Bdry- Tm: We have $\mathcal{B} = (\square : A)$. From $\Xi; \Gamma \vdash A$ type we obtain $\Theta; \Gamma \vdash I_*A \equiv J_*A$ using Theorem 3.17, and convert $\Theta; \Gamma \vdash e : I_*A$ to $\Theta; \Gamma \vdash e : J_*A$.

*Case* TT- Bdry- EqTy: We have $\mathcal{B} = (A \equiv B$ by $\square)$. From Theorem 3.18 we get $\Xi; \Gamma \vdash A$ type, hence $\Theta; \Gamma \vdash I_*A \equiv J_*A$ by Theorem 3.17. It follows similarly that $\Theta; \Gamma \vdash I_*B \equiv J_*B$. We may combine these with $\Theta; \Gamma \vdash I_*A \equiv I_*B$ using transitivity to derive $\Theta; \Gamma \vdash J_*A \equiv J_*B$.

*Case* TT- Bdry- EqTm: We have $\mathcal{B} = (s \equiv t : A)$. From Theorem 3.18 we get

$$\Xi; \Gamma \vdash A \text{ type}, \qquad \Xi; \Gamma \vdash s : A, \quad \text{and} \quad \Xi; \Gamma \vdash t : A.$$

and from Theorem 3.17

$$\Xi; \Gamma \vdash I_*A \equiv J_*A, \qquad \Xi; \Gamma \vdash I_*s \equiv J_*s : I_*A, \quad \text{and} \quad \Xi; \Gamma \vdash I_*t \equiv J_*t : I_*A.$$

Together with $\Xi; \Gamma \vdash I_*s \equiv I_*t : I_*A$ this is sufficient to derive $\Xi; \Gamma \vdash J_*s \equiv J_*t : J_*A$ using transitivity and conversions.

*Case* TT- Bdry- Abstr: We have $\mathcal{B} = (\{x{:}A\}\ \mathcal{B}')$ and $\Xi; \Gamma \vdash \{x{:}I_*A\}\ (I_*\mathcal{B}')\boxed{e}$. We use induction hypothesis and abstraction to derive $\Xi; \Gamma \vdash \{x{:}I_*A\}\ (J_*\mathcal{B}')\boxed{e}$ and then convert the abstracion to $J_*A$ using TT- Conv- Abstr. □

**Proposition 3.22** (Economic version of Definition 2.17) *In a finitary type theory, consider one of its object rules $R$*

$$\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_n{:}\mathcal{B}_n \Longrightarrow \mathcal{b}\boxed{e}.$$

*Given instantiations of its premises,*

$$I = \langle \mathsf{M}_1 \mapsto f_1, \ldots, \mathsf{M}_n \mapsto f_n \rangle \quad \textit{and} \quad J = \langle \mathsf{M}_1 \mapsto g_1, \ldots, \mathsf{M}_n \mapsto g_n \rangle,$$

*over* $\Theta; \Gamma$ *such that* $\vdash \Theta$ mctx *and* $\Theta \vdash \Gamma$ vctx*, the following closure rule is admissible:*

TT- Congr- Eco

$$\frac{\Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i} \qquad \text{for equation boundary } \mathcal{B}_i \qquad \Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i \equiv g_i} \qquad \text{for object boundary } \mathcal{B}_i}{\Theta; \Gamma \vdash (I_*\mathcal{b})\boxed{I_*e \equiv J_*e}}$$

***Proof*** We appeal to the congruence rule for $R$,

$$\frac{\begin{array}{ll}\Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i} & \text{for } i = 1, \ldots, n \\ \Theta; \Gamma \vdash (J_{(i)*}\mathcal{B}_i)\boxed{g_i} & \text{for } i = 1, \ldots, n \\ \Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{f_i \equiv g_i} & \text{for object boundary } \mathcal{B}_i \\ \Theta; \Gamma \vdash I_*B \equiv J_*B & \text{if } \mathcal{b} = (\Box : B)\end{array}}{\Theta; \Gamma \vdash (I_*\mathcal{b})\boxed{I_*e \equiv J_*e}}$$

whose premises are derived as follows.

The equational premises of the first row are given, while the object premises follow from the corresponding equational premises in TT- Congr- Eco by Theorem 3.18.

The second row of premises is more challenging. First, for each object premise, applying Theorem 3.18 to the corresponding equational premise in TT- Congr- Eco yields $\Theta; \Gamma \vdash (I_{(i)*}\mathcal{B}_i)\boxed{g_i}$ which is then converted to $\Theta; \Gamma \vdash (J_{(i)*}\mathcal{B}_i)\boxed{g_i}$ with the aid of Lemma 3.21. For an equational premise, we again use Lemma 3.21, except that we apply it to the corresponding equational premise in the first row, noting that in this case $f_i$ and $g_i$ are the same.

The third row of premises is given. The last premise, when present, follows by Theorem 3.17 from the fact that $R$ is finitary. □

**Theorem 3.24** (Inversion) *If a standard type theory derives an object judgement then there is a derivation of this judgement which concludes with precisely one of the following rules:*

1. *the variable rule* TT- Var*,*
2. *the metavariable rule* TT- Meta*,*
3. *an instantiation of a symbol rule,*
4. *the abstraction rule* TT- Abstr*,*
5. *the term conversion rule* TT- Conv- Tm *of the form*

$$\frac{\Theta; \Gamma \vdash t : \tau_{\Theta;\Gamma}(t) \qquad \Theta; \Gamma \vdash \tau_{\Theta;\Gamma}(t) \equiv A}{\Theta; \Gamma \vdash t : A}$$

*where* $\tau_{\Theta;\Gamma}(t) \neq A$.

***Proof*** We proceed by induction on the derivation $\Gamma; \Theta \vdash \mathcal{J}$. If the derivation concludes with TT- Var, TT- Meta, a symbol rule, or TT- Abstr, then it already has the desired form. The remaining case is a derivation $D$ ending with a term conversion rule

$$\frac{\dfrac{D_1}{\Theta; \Gamma \vdash t : A} \qquad \dfrac{D_2}{\Theta; \Gamma \vdash A \equiv B}}{\Theta; \Gamma \vdash t : B}$$

By induction hypothesis we may invert $D_1$ and obtain a derivation $D'$ of $\Theta; \Gamma \vdash t : A$ as in the statement of the theorem:

1. If $D'$ ends with TT- Var, TT- Meta or a term symbol rule then $A = \tau_{\Theta;\Gamma}(t)$. Either $\tau_{\Theta;\Gamma}(t) = B$ and we use $D'$, or $\tau_{\Theta;\Gamma}(t) \neq B$ and we use $D$.
2. If $D'$ concludes with a term conversion

$$\dfrac{\dfrac{D_1'}{\Theta; \Gamma \vdash t : \tau_{\Theta;\Gamma}(t)} \qquad \dfrac{D_2'}{\Theta; \Gamma \vdash \tau_{\Theta;\Gamma}(t) \equiv A}}{\Theta; \Gamma \vdash t : A}$$

there are again two cases. If $\tau_{\Theta;\Gamma}(t) = B$ we use $D_1'$, otherwise we combine $\tau_{\Theta;\Gamma}(t) \equiv A$ and $A \equiv B$ by transitivity and conversion:

$$\dfrac{\dfrac{D_1'}{\Theta; \Gamma \vdash t : \tau_{\Theta;\Gamma}(t)} \qquad \dfrac{\dfrac{D_2'}{\Theta; \Gamma \vdash \tau_{\Theta;\Gamma}(t) \equiv A} \qquad \dfrac{D_2}{\Theta; \Gamma \vdash A \equiv B}}{\Theta; \Gamma \vdash \tau_{\Theta;\Gamma}(t) \equiv B}}{\Theta; \Gamma \vdash t : B}$$

□

## A.2 Proofs of Meta-theorems About Context-Free Type Theories

This section provides missing proofs from Sect. 5.

**Lemma 5.2** *If a context-free raw type theory derives*

$$\vdash \{x_1{:}A_1\} \cdots \{x_n{:}A_n\}\ \mathcal{J} \qquad \text{and}$$
$$\vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] \qquad \text{for } i = 1, \ldots, n$$

*then it derives* $\vdash \mathcal{J}[\vec{t}/\vec{x}]$.

***Proof*** We may invert the derivation of $\vdash \{\vec{x}{:}\vec{A}\}\ \mathcal{J}$ to obtain a series of applications of CF- Abstr, yielding types $A_1', \ldots, A_n'$ and (suitably fresh) free variables $\mathsf{a}_1^{A_1'}, \ldots, \mathsf{a}_n^{A_n'}$ where, for $i = 1, \ldots, n$,

$$A_i' = A_i[\mathsf{a}_1^{A_1'}/x_1, \ldots, \mathsf{a}_{i-1}^{A_{i-1}'}/x_{i-1}] \quad \text{and} \quad \vdash A_i' \ \mathsf{type}.$$

At the top of the abstractions sits a derivation $D$ of the judgement

$$\mathcal{J}[\mathsf{a}_1^{A_1'}/x_1, \ldots, \mathsf{a}_n^{A_n'}/x_n].$$

The proof proceeds by induction on the derivation $D$, i.e. we only ever apply the induction hypotheses to derivations that have a series of abstractions, and on the top a derivation that is structurally smaller than $D$. Let us write

$$\theta = [\mathsf{a}_1^{A_1'}/x_1, \ldots, \mathsf{a}_n^{A_n'}/x_n],$$
$$\zeta = [x_n/\mathsf{a}_n^{A_n'}, \ldots, x_1/\mathsf{a}_1^{A_1'}],$$
$$\tau = [t_1/x_1, \ldots, t_n/x_n].$$

*Case* CF- Var: Suppose the derivation ends with the variable rule

$$\dfrac{}{\vdash \mathsf{b}^B : B}$$

If $\mathsf{b}^B$ is one of $\mathsf{a}_i^{A_i'}$ then $\mathcal{J} = \{\vec{x}{:}\vec{A}\}\ x_i : A_i$, hence $\mathcal{J}\tau = (t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}])$, which is derivable by assumption. If $\mathsf{b}^B$ is none of $\mathsf{a}_i^{A_i}$'s then $\mathsf{a}_i^{A_i} \notin \mathsf{fv}(B)$ by freshness, hence $\mathcal{J}\tau = (\mathsf{b}^B : B)$, so we may reuse the same variable rule.

*Case* CF- ABSTR: Suppose the derivation ends with an abstraction

$$\frac{\vdash (A_{n+1}\theta)\ \mathsf{type} \qquad \mathsf{a}_{n+1}^{A_{n+1}\theta} \notin \mathsf{fv}(\mathcal{J}'\theta) \qquad (\mathcal{J}'\theta)[\mathsf{a}_{n+1}^{A_{n+1}\theta}/x_{n+1}]}{\vdash \{x_{n+1}{:}A_{n+1}\theta\}\ (\mathcal{J}'\theta)}$$

We extend the substitution by $t_{n+1} = \mathsf{a}_{n+1}^{A_{n+1}\tau}$ and apply the induction hypothesis to the abstracted derivation of the right-hand premise, whose conclusion is $\{\vec{x}{:}\vec{A}\}\{x_{n+1}{:}A_{n+1}\}\ \mathcal{J}'$, to obtain $\vdash\ \mathcal{J}'[\tau, t_{n+1}/x_{n+1}]$. We may abstract $\mathsf{a}_{n+1}^{A_{n+1}\tau}$ to get the desired judgement $\vdash \{x_{n+1}{:}A_{n+1}\tau\}\ (\mathcal{J}'\tau)$.

*All other cases* The remaining cases all follow the same pattern: abstract the premises, apply the induction hypotheses to them, and conclude with the same rule. We demonstrate how this works in case of $D$ ending with an instance of a specific rule $R = (\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow \mathcal{b}\boxed{e})$ instantiated with $I = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto e_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto e_n \rangle$:

$$\frac{\vdash (I_{(i)*}\mathcal{B}_i)\boxed{e_i} \quad \text{for } i = 1, \ldots, n \qquad \vdash I_*\mathcal{b}}{\vdash I_*(\mathcal{b}\boxed{e})}$$

Define the instantiation $J$ of the premises of $R$ by $J(\mathsf{M}_i) = e_i' = (e_i\zeta)\tau$. Note that $\vdash (I_*(\mathcal{b}\boxed{e}))\tau$ equals $\vdash J_*(\mathcal{b}\boxed{e})$, therefore we may derive it by $J_*R$. The last premise of $J_*R$ is $\vdash (I_*\mathcal{b})\tau$, and it follows by Lemma 5.3 applied to the last premise of $I_*R$. For $i = 1, \ldots, n$, abstract $\vdash (I_{(i)*}\mathcal{B}_i)\boxed{e_i}$ to

$$\vdash \{\vec{x}{:}\vec{A}\}\ ((I_{(i)*}\mathcal{B}_i)\boxed{e_i})\zeta$$

and apply the induction hypothesis to derive $\vdash\ (((I_{(i)*}\mathcal{B}_i)\boxed{e_i})\zeta)\tau$, which equals $\vdash (((I_{(i)*}\mathcal{B}_i)\zeta)\tau)\boxed{(e_i\zeta)\tau}$ and because $\mathcal{B}_i$ does not contain any free variables, also to $\vdash (J_{(i)*}\mathcal{B}_i)\boxed{e_i'}$. □

**Theorem 5.5** (Context-free presuppositivity) *If a context-free raw type theory derives* $\vdash \mathcal{B}\boxed{e}$ *and* $\mathcal{B}\boxed{e}$ *has well-typed annotations, then it derives* $\vdash \mathcal{B}$.

***Proof*** The proof proceeds by induction on the number of metavariables appearing in the judgement and the derivation of $\vdash \mathcal{B}\boxed{e}$. That is, each appeal to the induction hypothesis reduces the number of metavariables, or is applied to a subderivation.

*Case* CF- VAR: Immediate, by the well-typedness of annotations.

*Case* CF- META: Immedate as the desired judgement is a premise of the rule.

*Case* CF- META- CONGR- TM: Suppose $\mathcal{B} = \{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \square : B$ and consider a derivation ending with the metavariable congruence rule

$$\dfrac{\begin{array}{l}\text{for } k = 1, \ldots, m : \\ \vdash s_k : A_k[\vec{s}_{(k)}/\vec{x}_{(k)}] \\ \vdash t_k : A_k[\vec{t}_{(k)}/\vec{x}_{(k)}] \\ \lfloor t_k \rfloor = \lfloor t_k' \rfloor \\ \vdash s_k \equiv t_k' : A_k[\vec{s}_{(k)}/\vec{x}_{(k)}] \text{ by } \alpha_k \\ \vdash v : B[\vec{s}/\vec{x}] \qquad \lfloor \mathsf{M}^{\mathcal{B}}(\vec{t}) \rfloor = \lfloor v \rfloor \\ \beta \text{ suitable}\end{array}}{\vdash \mathsf{M}^{\mathcal{B}}(\vec{s}) \equiv v : B[\vec{s}/\vec{x}] \text{ by } \beta}$$

The presuppositions are derived as follows:

- $\vdash B[\vec{s}/\vec{x}]\ \mathsf{type}$ follows by CF- SUBST from $\vdash \vec{x}{:}\vec{A}\ B\ \mathsf{type}$, which in turn follows by inversion on $\vdash \mathcal{B}$.
- $\vdash \mathsf{M}^{\mathcal{B}}(\vec{s}) : B[\vec{s}/\vec{x}]$ follows by CF- META.
- $v : B[\vec{s}/\vec{x}]$ is a premise.

*Case* CF- ABSTR: Consider an abstraction

$$\dfrac{\vdash A\ \mathsf{type} \qquad \mathsf{a}^A \notin \mathsf{fv}(\mathcal{B}\boxed{e}) \qquad \vdash (\mathcal{B}\boxed{e})[\mathsf{a}^A/x]}{\vdash \{x{:}A\}\ \mathcal{B}\boxed{e}}$$

By induction hypothesis on the last premise, we obtain $\vdash \mathcal{B}[\mathsf{a}^A/x]$ after which we apply CF- BDRY- ABSTR.

*Case of a specific rule*: Immediate, as the well-formedness of the boundary is a premise.

*Case of a congruence rule*: Consider a congruence rulles associated with an object rule $R$ and instantiated with $I$ and $J$, as in Definition 4.8.

If $R$ concludes with $\vdash A\ \mathsf{type}$, the presuppositions are $\vdash I_* A\ \mathsf{type}$ and $\vdash J_* A\ \mathsf{type}$, which are derivable by $I_* R$ and $J_* R$, respectively.

If $R$ concludes with $\vdash t : A$, the presuppositions are $\vdash I_* A\ \mathsf{type}$, $\vdash I_* t : I_* A$, and $\vdash t' : I_* A$. We derive the first one by applying the induction hypothesis to the premise $\vdash t' : I_* A$, the second one by $I_* R$, while the third one is a premise.

*Cases* CF- EQTY- REFL, CF- EQTY- SYM, CF- EQTY- TRANS, CF- EQTM- REFL, CF- EQTM- SYM, CF- EQTM- TRANS: These are all dispensed with straightforward appeals to the induction hypotheses.

*Case* CF- CONV- TM: Consider a term conversion

$$\dfrac{\begin{array}{c}\vdash t : A \qquad \vdash A \equiv B \text{ by } \alpha \\ \mathsf{asm}(t, A, B, \alpha) = \mathsf{asm}(t, B, \beta)\end{array}}{\vdash \kappa(t, \beta) : B}$$

By induction hypothesis for the second premise, $\vdash B\ \mathsf{type}$.

*Case* CF- CONV- EQTM: Consider a term equality conversion

$$\dfrac{\begin{array}{c}\vdash s \equiv t : A \text{ by } \alpha \qquad \vdash A \equiv B \text{ by } \beta \\ \mathsf{asm}(s, A, B, \beta) = \mathsf{asm}(s, B, \gamma) \\ \mathsf{asm}(t, A, B, \beta) = \mathsf{asm}(t, B, \delta)\end{array}}{\vdash \kappa(s, \gamma) \equiv \kappa(t, \delta) : B \text{ by } \alpha}$$

As in the previous case, the induction hypothesis for the second premise provides $\vdash B\ \mathsf{type}$. The induction hypothesis for the first premise yields

$$\vdash s : A \qquad \text{and} \qquad \vdash t : A$$

We may convert these to $\vdash \kappa(s,\gamma) : B$ and $\vdash \kappa(t,\delta) : B$ using the second premise. □

**Lemma 5.6** *If a context-free raw type theory derives*

$$\vdash \{x_1{:}A_1\} \cdots \{x_n{:}A_n\}\ \mathcal{J}$$

*where* $\{\vec{x}{:}\vec{A}\}\ \mathcal{J}$ *has well-typed annotations, and for* $i = 1, \ldots, n$

$$\begin{aligned} &\vdash s_i : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}] \\ &\vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] \\ &\vdash s_i \equiv t_i' : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}]\ \mathsf{by}\ \alpha_i \qquad \text{and } \lfloor t_i' \rfloor = \lfloor t_i \rfloor. \end{aligned} \tag{5.1}$$

*then:*

1. *if* $\mathcal{J} = (\{\vec{y}{:}\vec{B}\}\ C\ \mathsf{type})$ *then there are* $\gamma$ *and* $C'$ *such that* $\lfloor C[\vec{t}/\vec{x}] \rfloor = \lfloor C' \rfloor$,
$$\vdash \{\vec{y}{:}\vec{B}[\vec{s}/\vec{x}]\}\ C[\vec{s}/\vec{x}] \equiv C'\ \mathsf{by}\ \gamma,$$
2. *if* $\mathcal{J} = (\{\vec{y}{:}\vec{B}\}\ u : C)$ *then there are* $\delta$ *and* $u'$ *such that* $\lfloor u[\vec{t}/\vec{x}] \rfloor = \lfloor u' \rfloor$ *and*
$$\vdash \{\vec{y}{:}\vec{B}[\vec{s}/\vec{x}]\}\ u[\vec{s}/\vec{x}] \equiv u' : C[\vec{s}/\vec{x}]\ \mathsf{by}\ \delta.$$

*Furthermore, no extraneous assumptions are introduced by* $\gamma$, $C'$, $\delta$ *and* $u'$*:*

$$\mathsf{asm}(\{\vec{y}\}\gamma, \{\vec{y}\}C', \{\vec{y}\}\delta, \{\vec{y}\}u') \subseteq \mathsf{asm}(\vec{s}, \vec{t}, \vec{t}', \vec{\alpha}, \{\vec{x}{:}\vec{A}\}\ \mathcal{J}).$$

***Proof*** . As in the proof of Lemma 5.2, we invert the derivation of $\vdash \{\vec{x}{:}\vec{A}\}\ \mathcal{J}$ to obtain types $A_1', \ldots, A_n'$ and (suitably fresh) free variables $\mathsf{a}_1^{A_1'}, \ldots, \mathsf{a}_n^{A_n'}$ where, for $i = 1, \ldots, n$,

$$A_i' = A_i[\mathsf{a}_1^{A_1'}/x_1, \ldots, \mathsf{a}_{i-1}^{A_{i-1}'}/x_{i-1}] \quad \text{and} \quad \vdash A_i'\ \mathsf{type}$$

and a derivation $D$ of the judgement

$$\mathcal{J}[\mathsf{a}_1^{A_1'}/x_1, \ldots, \mathsf{a}_{i-1}^{A_n'}/x_n].$$

The proof proceeds by induction on the well-founded ordering of the rules, the number of metavariables, with a subsidiary induction on the derivation $D$. That is, each appeal to the induction hypotheses either decreases the number of metavariables appearing in the judgement, or descends to a subderivation of $D$. Let us write

$$\begin{aligned} \theta &= [\mathsf{a}_1^{A_1'}/x_1, \ldots, \mathsf{a}_{i-1}^{A_n'}/x_n], \\ \sigma &= [s_1/x_1, \ldots, s_n/x_n], \\ \tau &= [t_1/x_1, \ldots, t_n/x_n]. \end{aligned}$$

*Case* CF- VAR: Suppose the derivation ends with the variable rule

$$\frac{}{\vdash \mathsf{b}^B : B}$$

If $\mathsf{b}^B$ is one of $\mathsf{a}_i^{A_i'}$ then $\mathcal{J} = \{\vec{x}{:}\vec{A}\}\ x_i : A_i$, hence (2) is satisfied by (5.1). If $\mathsf{b}^B$ is none of $\mathsf{a}_i^{A_i}$'s then $\mathsf{a}_i^{A_i} \notin \mathsf{fv}(B)$ by freshness, hence (2) is satisfied by $\vdash \mathsf{b}^B \equiv \mathsf{b}^B : B$ by $\{\!\}$, which holds by CF- EqTm- Refl.

*Case* CF- Abstr: Suppose the derivation ends with an abstraction

$$\frac{\vdash (A_{n+1}\theta)\ \mathsf{type} \qquad \mathsf{a}_{n+1}^{A_{n+1}\theta} \notin \mathsf{fv}(\mathcal{J}\theta) \qquad (\mathcal{J}'\theta)[\mathsf{a}_{n+1}^{A_{n+1}\theta}/y]}{\vdash \{x_{n+1}{:}A_{n+1}\theta\}\ (\mathcal{J}'\theta)} \tag{A37}$$

We may abstract the first premise to $\vdash \{\vec{x}{:}\vec{A}\}\ A_{n+1}\ \mathsf{type}$, apply Lemma 5.2 to derive $\vdash A_{n+1}\tau\ \mathsf{type}$, and the induction hypothesis to obtain $\beta_{n+1}$ and $A'$ such that $\lfloor A_{n+1}\tau \rfloor = \lfloor A' \rfloor$,

$$\vdash A'\ \mathsf{type} \qquad \text{and} \qquad \vdash A_{n+1}\sigma \equiv A'\ \mathsf{by}\ \beta_{n+1}.$$

By CF- EqTy- Trans and CF- EqTy- Refl it follows that for some $\gamma_{n+1}$

$$\vdash A_{n+1}\sigma \equiv A_{n+1}\tau\ \mathsf{by}\ \gamma_{n+1}.$$

Let $\mathsf{a}_{n+1}^{A_{n+1}\sigma}$ be fresh, and define

$$s_{n+1} = t'_{n+1} = \mathsf{a}_{n+1}^{A_{n+1}\sigma}, \qquad t_{n+1} = \kappa(\mathsf{a}_{n+1}^{A_{n+1}\sigma}, \gamma_{n+1}), \quad \text{and} \quad \alpha_{n+1} = \{\!\}.$$

We may abstract the last premise of (A37) to

$$\vdash \{x_1{:}A_1\} \cdots \{x_{n+1}{:}A_{n+1}\}\ \mathcal{J}'$$

apply the induction hypothesis with the given $s_{n+1}$, $t_{n+1}$ and $t'_{n+1}$ to derive either (1) or (2), and abstract $\mathsf{a}_{n+1}^{A_{n+1}\sigma}$ to get the desired judgements.

*Case* CF- Meta: We consider the case of an object metavariable, and leave the easier case of a type metavariable to the reader. Let $\mathcal{B} = (\{\vec{y}{:}\vec{B}\}\ \square : C)$, and suppose the derivation ends with an application of the metavariable rule,

$$\frac{\begin{array}{l}\vdash u_j\theta : B_j[\vec{u}_{(j)}\theta/\vec{y}_{(j)}] \quad \text{for } j = 1, \ldots, m \\ \vdash \square : C[\vec{u}\theta/\vec{y}]\end{array}}{\vdash \mathsf{M}^{\mathcal{B}}(\vec{u}\theta) : C[\vec{u}\theta/\vec{y}]} \tag{A38}$$

For each $j = 1, \ldots, m$ we may abstract the premise of (A38) to

$$\vdash \{\vec{x}{:}\vec{A}\}\ u_j : B_j[\vec{u}_{(j)}/\vec{y}_{(j)}]$$

and apply Lemma 5.2, once with $\vec{s}$ and once with $\vec{t}$, to derive

$$\vdash u_j\sigma : B_j[(\vec{u}\sigma)_{(j)}/\vec{y}_{(j)}],$$
$$\vdash u_j\tau : B_j[(\vec{u}\tau)_{(j)}/\vec{y}_{(j)}],$$

where we took into account the fact that $B_j$ does not contain any bound variables. Also, by induction hypothesis there are $\delta_j$ and $u'_j$ such that $\lfloor u_j\tau \rfloor = \lfloor u'_j \rfloor$ and

$$\vdash u_j\sigma \equiv u'_j : B_j[(\vec{u}\sigma)_{(j)}/\vec{y}_{(j)}]\ \mathsf{by}\ \delta_j.$$

Next, we invert the last premise of (A38) and abstract it to $\vdash \{\vec{x}{:}\vec{A}\}\ C[\vec{u}/\vec{y}]\ \mathsf{type}$. By induction hypothesis we obtain $\delta'$ and $C'$ such that $\lfloor C' \rfloor = \lfloor C[\vec{u}\tau/\vec{y}] \rfloor$ and $\vdash C[\vec{u}\sigma/\vec{y}] \equiv C'$ by $\delta'$,

hence $\vdash C[\vec{u}\sigma/\vec{y}] \equiv C[\vec{u}\tau/\vec{y}]$ by $\delta''$ for some $\delta''$. Now (2) is satisfied, for some $\delta'''$

$$\vdash \kappa(\mathsf{M}^{\mathcal{B}}(\vec{u}\tau), \delta) : C[\vec{u}\sigma/\vec{y}],$$
$$\vdash \mathsf{M}^{\mathcal{B}}(\vec{u}\sigma) \equiv \kappa(\mathsf{M}^{\mathcal{B}}(\vec{u}\tau), \delta) : C[\vec{u}\sigma/\vec{y}] \text{ by } \delta'''$$

where the last judgement follows by the congruence rule for $\mathsf{M}^{\mathcal{B}}$.

*Case of a specific term rule*: Suppose the derivation ends with a specific rule $R = (\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_m^{\mathcal{B}_m} \Longrightarrow u : B)$ instantiated with $I' = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto e'_1, \ldots, \mathsf{M}_m^{\mathcal{B}_m} \mapsto e'_m \rangle$:

$$\frac{\vdash (I'_{(j)*}\mathcal{B}_j)\boxed{e'_j} \quad \text{for } j = 1, \ldots, n \qquad \vdash (\square : I'_* B)}{\vdash I'_* u : I'_* B}$$

Let $\zeta = [x_n/\mathsf{a}_n^{A'_n}, \ldots, x_1/\mathsf{a}_1^{A'_1}]$ be the abstraction that undoes $\theta$. Define $e_j = e'_j\zeta$ and $I = I'\zeta$, so that $e'_j = e_j\theta$ and $I' = I\theta$, which allows us to write the above judgement as

$$\frac{\vdash ((I_{(j)*}\mathcal{B}_j)\boxed{e_j})\theta \quad \text{for } j = 1, \ldots, n \qquad \vdash (\square : I_* B)\theta}{\vdash (I_* u)\theta : (I_* B)\theta}$$

We invert the last premise, abstract to $\vdash \{\vec{x}{:}\vec{A}\}\ I_* B$ type, and apply Lemma 5.2 to derive $\vdash (I_* B)\sigma$ type. Next, the induction hypothesis provides $\beta$ and $B'$ such that $\lfloor B' \rfloor = \lfloor (I_* B)\tau \rfloor$ and $\vdash (I_* B)\sigma \equiv B'$ by $\beta$. Therefore, we have $\beta'$ such that

$$\vdash (I_* B)\sigma \equiv (I_* B)\tau \text{ by } \beta'.$$

It suffices to show

$$\vdash (I_* u)\sigma \equiv \kappa((I_* u)\tau, \beta') : (I_* B)\sigma \text{ by } \delta$$

for a suitable $\delta$. This is precisely the conclusion of the congruence rule for $R$, so we derive its premises. For any $j = 1, \ldots, m$ we may abstract the $j$-th premise to

$$\vdash \{\vec{x}{:}\vec{A}\}\ (I_{(j)*}\mathcal{B}_j)\boxed{e_j}, \tag{A39}$$

and apply Lemma 5.2, once with $\vec{s}$ and once with $\vec{t}$, to derive

$$\vdash ((I\sigma)_{(j)*}\mathcal{B}_j)\boxed{e_j\sigma} \quad \text{and} \quad \vdash ((I\tau)_{(j)*}\mathcal{B}_j)\boxed{e_j\tau}.$$

For each object premise with boundary $\mathcal{B}_j$, the remaining premises are provided precisely by the induction hypotheses.

*Case of a specific type rule*: Suppose the derivation ends with a specific rule $R = (\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_m^{\mathcal{B}_m} \Longrightarrow B \text{ type})$ instantiated with $I' = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto e'_1, \ldots, \mathsf{M}_m^{\mathcal{B}_m} \mapsto e'_m \rangle$:

$$\frac{\vdash (I'_{(j)*}\mathcal{B}_j)\boxed{e'_j} \quad \text{for } j = 1, \ldots, n \qquad \vdash \square \text{ type}}{\vdash I'_* B \text{ type}}$$

With $\zeta$ and $I$ as in the previous case, we may write the above as

$$\frac{\vdash ((I_{(j)*}\mathcal{B}_j)\boxed{e_j})\theta \quad \text{for } j = 1, \ldots, n}{\vdash (I_* B)\theta \text{ type}}$$

where we elided the trivial boundary premise. It suffices to find a suitable $\gamma$ such that $\vdash (I_* B)\sigma \equiv (I_* B)\tau$ by $\gamma$, which is precisely the conclusion of the congruence rule for $R$, whose premises are derived as in the previous case.

*Case* CF- Conv- Tm: Suppose the derivation ends with an application of the conversion rule

$$\frac{\vdash u\theta : B\theta \qquad \vdash B\theta \equiv C\theta \text{ by } \beta\theta}{\vdash \kappa(u\theta, \beta\theta \cup \mathsf{asm}(B\theta)) : C\theta}$$

We abstract the first premise to $\vdash \{\vec{x}{:}\vec{A}\}\ u : B$ and apply the induction hypothesis to obtain $\delta$ and $u'$ such that $\lfloor u' \rfloor = \lfloor u\tau \rfloor$ and

$$\vdash u\sigma \equiv u' : B\sigma \text{ by } \delta.$$

We abstract the second premise to $\vdash \{\vec{x}{:}\vec{A}\}\ B \equiv C$ by $\beta$, apply Lemma 5.2 to derive $\vdash B\sigma \equiv C\sigma$ by $\beta\sigma$, and use CF- Conv- EqTm to conclude, for suitable $\gamma$ and $\gamma'$,

$$\vdash \kappa(u\sigma, \gamma) \equiv \kappa(u', \gamma') : C\sigma \text{ by } \delta.$$

□

**Theorem 5.7** *In a context-free raw type theory, the following closure rules are admissible:*

CF- Subst- EqTy

$$\frac{\begin{array}{ll}\vdash \{\vec{x}{:}\vec{A}\}\{\vec{y}{:}\vec{B}\}\ C \text{ type} & \\ \vdash s_i : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}] & \text{for } i = 1, \ldots, n \\ \vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] & \text{for } i = 1, \ldots, n \\ \lfloor t_i \rfloor = \lfloor t_i' \rfloor & \text{for } i = 1, \ldots, n \\ \vdash s_i \equiv t_i' : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}] \text{ by } \alpha_i & \text{for } i = 1, \ldots, n \\ & \beta \text{ suitable}\end{array}}{\vdash \{\vec{y}{:}\vec{B}[\vec{s}/\vec{x}]\}\ C[\vec{s}/x] \equiv C[\vec{t}/x] \text{ by } \beta}$$

CF- Subst- EqTm

$$\frac{\begin{array}{ll}\vdash \{\vec{x}{:}\vec{A}\}\{\vec{y}{:}\vec{B}\}\ u : C & \\ \vdash s_i : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}] & \text{for } i = 1, \ldots, n \\ \vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] & \text{for } i = 1, \ldots, n \\ \lfloor t_i \rfloor = \lfloor t_i' \rfloor & \text{for } i = 1, \ldots, n \\ \vdash s_i \equiv t_i' : A_i[\vec{s}_{(i)}/\vec{x}_{(i)}] \text{ by } \alpha_i & \text{for } i = 1, \ldots, n \\ & \beta \text{ suitable}\end{array}}{\vdash \{\vec{y}{:}\vec{B}[\vec{s}/\vec{x}]\}\ u[\vec{s}/\vec{x}] \equiv \kappa(u[\vec{t}/\vec{x}], \beta) : C[\vec{s}/x] \text{ by } \beta}$$

***Proof*** Lemma 5.6 applied to the premises of CF- Subst- EqTy provides $\gamma$ and $C'$ such that $\lfloor C[\vec{t}/\vec{x}] \rfloor = \lfloor C' \rfloor$ and

$$\vdash \{\vec{y}{:}\vec{B}[\vec{s}/\vec{x}]\}\ C[\vec{s}/\vec{x}] \equiv C' \text{ by } \gamma. \tag{A40}$$

We would like to replace $C'$ in the right-hand side with $C[\vec{t}/\vec{x}]$, which we can so long as

$$\vdash \{\vec{y}{:}\vec{B}[\vec{s}/\vec{x}]\}\ C' \text{ type} \quad \text{and} \quad \vdash \{\vec{y}{:}\vec{B}[\vec{s}/\vec{x}]\}\ C[\vec{t}/\vec{x}].$$

The first judgement holds by Theorem 5.5 applied to (A40) under the abstraction, while the second one is a substitution instance of the first premise. This establishes derivability of CF- SUBST- EQTY.

In case of CF- SUBST- EQTM the same lemma yields $\delta$ and $u'$ such that $\lfloor u[\vec{t}/\vec{x}] \rfloor = \lfloor u' \rfloor$ and

$$\vdash \{\vec{y}{:}\vec{B}[\vec{s}/\vec{x}]\}\ u[\vec{s}/\vec{x}] \equiv u' : C[\vec{s}/x] \text{ by } \delta.$$

We would like to replace $u'$ with a converted $u[\vec{t}/\vec{x}]$, which we can by an argument similar to the one above. □

**Theorem 5.8** (Context-free admissibility of instantiation) *In a raw type theory, if* $\vdash \mathcal{J}$ *is derivable, it has well-typed annotations, and* $I$ *is a derivable instantiation such that* $\mathsf{mv}(\mathcal{J}) \subseteq |I|$*, then* $\vdash I_*\mathcal{J}$ *is derivable, and similarly for boundaries.*

***Proof*** . We proceed by induction on the derivation of $\vdash \mathcal{J}$. We only devote attention to the metavariable and abstraction rules, as all the other cases are straightforward. Suppose $I = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto e_1, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \mapsto e_n \rangle$.

*Case* CF- META: Consider an application of the metavariable rule for $\mathsf{M}_i^{\mathcal{B}_i}$ with $\mathcal{B}_i = (\{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \mathcal{b})$ and $e_i = \{\vec{x}\}e$:

$$\frac{\vdash t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] \quad \text{for } i = 1, \ldots, n \qquad \vdash \mathcal{b}[\vec{t}/\vec{x}]}{\vdash ((\mathcal{b}[\vec{t}/\vec{x}])\boxed{\mathsf{M}_i^{\mathcal{B}_i}(\vec{t})})}$$

Because $I_*((\mathcal{b}[\vec{t}/\vec{x}])\boxed{\mathsf{M}_i^{\mathcal{B}_i}(\vec{t})}) = ((I_*\mathcal{b})[I_*\vec{t}/\vec{x}])\boxed{e[I_*\vec{t}/\vec{x}]}$ we need to derive

$$\vdash ((I_*\mathcal{b})[I_*\vec{t}/\vec{x}])\boxed{e[I_*\vec{t}/\vec{x}]}. \tag{A41}$$

Because $I$ is derivable, we know that $\vdash \{\vec{x}{:}I_{(i)*}\vec{A}\}\ (I_{(i)*}\mathcal{b})\boxed{e}$. By induction hypothesis $\vdash I_{(i)*}t_j : (I_{(i)*}A_j)[I_{(i)*}\vec{t}_{(j)}/\vec{x}_{(j)}])$ for each $j = 1, \ldots, m$, so by Lemma 5.2 we derive $\vdash ((I_{(i)*}\mathcal{b})\boxed{e})[I_{(i)*}\vec{t}/\vec{x}]$, which coincides with (A41).

*Case* CF- META- CONGR- TY: We consider the congruence rule for types only. Suppose the derivation ends with an application of the congruence rule for $\mathsf{M}_i^{\mathcal{B}_i}$ with $\mathcal{B}_i = (\{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \square\ \mathsf{type})$ and $e_i = \{\vec{x}\}B$:

$$\frac{\begin{array}{l}\text{for } k = 1, \ldots, m : \\ \vdash s_k : A[\vec{s}_{(k)}/\vec{x}_{(k)}] \\ \vdash t_k : A[\vec{t}_{(k)}/\vec{x}_{(k)}] \\ \lfloor t_k \rfloor = \lfloor t'_k \rfloor \\ \vdash s_k \equiv t'_k : A[\vec{s}_{(k)}/\vec{x}_{(k)}] \text{ by } \alpha_k \\ \beta \text{ suitable}\end{array}}{\vdash \mathsf{M}_i^{\mathcal{B}_i}(\vec{s}) \equiv \mathsf{M}_i^{\mathcal{B}_i}(\vec{t}) \text{ by } \beta}$$

Because $I$ is derivable, we know that $\vdash \{\vec{x}{:}I_{(i)*}\vec{A}\}\ B\ \mathsf{type}$, hence Lemma 5.6 applies.

*Case* CF- ABSTR: Suppose the derivation ends with an abstraction

$$\frac{\vdash A\ \mathsf{type} \qquad \mathsf{a}^A \notin \mathsf{fv}(\mathcal{J}) \qquad \vdash \mathcal{J}[\mathsf{a}^A/x]}{\vdash \{x{:}A\}\ \mathcal{J}}$$

Without loss of generality we may assume that $\mathsf{a}^{I_* A} \notin \mathsf{fv}(I_* \mathcal{J})$. (If not, rename $\mathsf{a}$ to a fresh symbol.) We may apply the induction hypotheses to both premises and get

$$\vdash I_* A\ \mathsf{type} \qquad \text{and} \qquad \vdash (I_* \mathcal{J})[\mathsf{a}^{I_* A}/x].$$

and derive the desired judgement $\vdash \{x{:}I_* A\}\ I_* \mathcal{J}$ by abstracting $\mathsf{a}^{I_* A}$ in the right-hand judgement. □

**Lemma 5.13** *If a context-free standard type theory derives* $\vdash t : A$ *then*

1. *it derives* $\vdash \mathfrak{s}(t) : \tau(t)$ *by an application of* CF- Var, CF- Meta, *or an instantiation of a term symbol rule, and*
2. *it derives* $\vdash \tau(t) \equiv A\ \mathsf{by}\ \mathfrak{r}(t)$.

***Proof*** We proceed by induction on the derivation of $\vdash t : A$.

*Cases* CF- Var, CF- Meta, *and symbol rule*: In these cases $t = \mathfrak{s}(t)$ and $\tau(t) = A$, so we already have $\vdash \mathfrak{s}(t) : \tau(t)$, while $\vdash \tau(t) \equiv A\ \mathsf{by}\ \{\!\}$ holds by reflexivity.

*Case* CF- Conv- Tm: Consider a derivation ending with a conversion

$$\frac{\vdash t : B \qquad \vdash B \equiv A\ \mathsf{by}\ \beta}{\vdash \kappa(t, \alpha) : A}$$

where $\mathsf{asm}(t, B, A, \beta) = \mathsf{asm}(t, A, \alpha)$. By induction hypothesis for the first premise we obtain $\vdash \mathfrak{s}(t) : \tau(t)$ and $\vdash \tau(t) \equiv B\ \mathsf{by}\ \mathfrak{r}(t)$, derived by one of the desired rules. Because $\mathfrak{s}(t) = \mathfrak{s}(\kappa(t, \alpha))$ and $\tau(t) = \tau(\kappa(t, \alpha))$, the first claim is established. For the second one, we apply CF- EqTy- Trans like this:

$$\frac{\vdash \tau(t) \equiv B\ \mathsf{by}\ \mathfrak{r}(t) \qquad \vdash B \equiv A\ \mathsf{by}\ \beta}{\vdash \tau(t) \equiv A\ \mathsf{by}\ \mathfrak{r}(t) \cup \alpha}$$

Suitability of $\mathfrak{r}(t) \cup \alpha$ is implied by $\mathsf{asm}(\tau(t), \mathfrak{r}(t)) = \mathsf{asm}(t)$:

$$\begin{aligned} \mathsf{asm}(\tau(t), B, \mathfrak{r}(t), A, \beta) &= \mathsf{asm}(t, B, A, \beta) \\ &= \mathsf{asm}(t, A, \alpha) \\ &= \mathsf{asm}(\tau(t), A, \mathfrak{r}(t), \alpha). \end{aligned}$$

□

**Theorem 5.17** (Boundary conversion) *In a context-free raw theory, if* $\vdash \mathcal{B}_1$, $\vdash \mathcal{B}_2$, $\vdash \mathcal{B}_1[e_1]$ *and* $\lfloor \mathcal{B}_1 \rfloor = \lfloor \mathcal{B}_2 \rfloor$ *then there is* $e_2$ *such that* $\vdash \mathcal{B}_2[e_2]$, $\mathsf{asm}(e_2) \subseteq \mathsf{asm}(\mathcal{B}_1[e_1])$ *and* $\lfloor e_1 \rfloor = \lfloor e_2 \rfloor$.

***Proof*** We proceed by induction on the derivation of $\vdash \mathcal{B}_1$.

*Case* CF- Bdry- Ty: If $\mathcal{B}_1 = (\square\ \mathsf{type})$ then $\mathcal{B}_2 = (\square\ \mathsf{type})$ and we may take $e_2 = e_1$.

*Case* CF- Bdry- Tm: If $\mathcal{B}_1 = (\square : A_1)$ then $\mathcal{B}_2 = (\square : A_2)$ and $\lfloor A_1 \rfloor = \lfloor A_2 \rfloor$, therefore $\vdash A_1 \equiv A_2\ \mathsf{by}\ \{\!\}$ by CF- EqTy- Refl. We may take $e_2 = \kappa(e_1, \mathsf{asm}(A_1) \backslash \mathsf{asm}(A_2))$ and derive $\vdash e_2 : A_2$ by CF- Conv- Tm.

*Case* CF- Bdry- EqTy: If $\mathcal{B}_1 = (A_1 \equiv B_1\ \mathsf{by}\ \square)$ then $\mathcal{B}_2 = (A_2 \equiv B_2\ \mathsf{by}\ \square)$, $\lfloor A_1 \rfloor = \lfloor A_2 \rfloor$ and $\lfloor B_1 \rfloor = \lfloor B_2 \rfloor$. By CF- EqTy- Refl we obtain $\vdash A_2 \equiv A_1\ \mathsf{by}\ \{\!\}$ and $\vdash B_1 \equiv B_2\ \mathsf{by}\ \{\!\}$. We take $e_2 = (e_1 \cup \mathsf{asm}(A_1) \cup \mathsf{asm}(B_1)) \backslash (\mathsf{asm}(A_2) \cup \mathsf{asm}(B_2))$ and derive $\vdash A_2 \equiv B_2\ \mathsf{by}\ e_2$ by two applications of CF- EqTy- Trans.

*Case* CF- BDRY- EQTM: If $\mathcal{B}_1 = (s_1 \equiv t_1 : A_1 \text{ by } \square)$ then $\mathcal{B}_2 = (s_2 \equiv t_2 : A_2 \text{ by } \square)$, $\lfloor s_1 \rfloor = \lfloor s_2 \rfloor$, $\lfloor t_1 \rfloor = \lfloor t_2 \rfloor$ and $\lfloor A_1 \rfloor = \lfloor A_2 \rfloor$. By CF- EQTY- REFL we obtain $\vdash A_1 \equiv A_2$ by $\{\!\}$, then by CF- CONV- EQTM

$$\vdash \kappa(s_1, \gamma) \equiv \kappa(t_1, \delta) : A_2 \text{ by } e_1$$

where $\gamma = \mathsf{asm}(A_1) \backslash \mathsf{asm}(s_1, A_2)$ and $\delta = \mathsf{asm}(A_1) \backslash \mathsf{asm}(t_1, A_2)$. Next, by reflexivity

$$\vdash s_2 \equiv \kappa(s_1, \gamma) : A_2 \text{ by } \{\!\}$$
$$\vdash \kappa(t_1, \delta) \equiv t_2 : A_2 \text{ by } \{\!\}$$

We may chain these together by transitivity to derive

$$\vdash s_2 \equiv t_2 : A_2 \text{ by } e_2$$

where $e_2 = \mathsf{asm}(e_1, s_1, t_1, A_1) \backslash \mathsf{asm}(s_2, t_2, A_2)$.

*Case* CF- BDRY- ABSTR: If $\mathcal{B}_1 = (\{x{:}A_1\}\ \mathcal{B}_1')$ then $e_1 = \{x\}e_1'$, $\mathcal{B}_2 = \{x{:}A_2\}\ \mathcal{B}_2'$, $\lfloor A_1 \rfloor = \lfloor A_2 \rfloor$, and $\lfloor \mathcal{B}_1' \rfloor = \lfloor \mathcal{B}_2' \rfloor$. There is $\mathsf{a}^{A_2} \notin \mathsf{fv}(\mathcal{B}_2')$ such that $\vdash \mathcal{B}_2'[\mathsf{a}^{A_2}/x]$. We may apply Lemma 5.2 to $\vdash \{x{:}A_1\}\ \mathcal{B}_1' \boxed{e_1'}$ and $\vdash \kappa(\mathsf{a}^{A_2}, \{\!\}) : A_1$ to derive

$$\vdash (\mathcal{B}_1'[\kappa(\mathsf{a}^{A_2}, \{\!\})/x]) \boxed{e_1'[\kappa(\mathsf{a}^{A_2}, \{\!\})/x]}.$$

By CF- BDRY- SUBST we have $\vdash \mathcal{B}_1'[\kappa(\mathsf{a}^{A_2}, \{\!\})/x]$, hence we may apply the induction hypothesis to obtain $e_2''$ such that $\lfloor e_2'' \rfloor = \lfloor e_1'[\kappa(\mathsf{a}^{A_2}, \{\!\})/x] \rfloor$, $\mathsf{asm}(e_2'') \subseteq \mathsf{asm}(\mathcal{B}_1'[\kappa(\mathsf{a}^{A_2}, \{\!\})/x])$, and $\vdash (\mathcal{B}_2'[\mathsf{a}^{A_2}/x]) \boxed{e_2''}$. Set $e_2' = e_2''[x/\mathsf{a}^{A_2}]$ and apply CF- BDRY- ABSTR to derive $\vdash \{x{:}A_2\}\ \mathcal{B}_2' \boxed{e_2'}$. Thus we may take $e_2 = \{x\}e_2'$. □

### A.3 Proofs of Theorems About Translation Betweeen tt- and cf-Type Theories

This section provides missing proofs from Sect. 6.

**Theorem 6.5** (Translation from finitary cf- to tt-theories)

1. *The translation of a finitary cf-theory is finitary.*
2. *Suppose $T$ is a finitary cf-theory whose translation $T_{\mathrm{tt}}$ is also finitary. Let $\Theta; \Gamma$ be tt-context such that $\vdash_{T_{\mathrm{tt}}} \Theta\ \mathsf{mctx}$ and $\Theta \vdash_{T_{\mathrm{tt}}} \Gamma\ \mathsf{vctx}$. If $\vdash_T \mathcal{J}$ and $\Theta; \Gamma$ is suitable for $\mathcal{J}$, then $\Theta; \Gamma \vdash_{T_{\mathrm{tt}}} \lfloor \mathcal{J} \rfloor$.*
3. *With $T$, $\Theta; \Gamma$ as in (2), if $\vdash_T \mathcal{B}$ and $\Theta; \Gamma$ is suitable for $\mathcal{B}$ then $\Theta; \Gamma \vdash_{T_{\mathrm{tt}}} \lfloor \mathcal{B} \rfloor$.*

***Proof*** We proceed by mutual structural induction on all three statements. To prove statement (1), consider a finitary cf-theory $T = (R_i)_{i \in I}$, and let $(I, \prec)$ be a well-founded order witnessing the finitary character of $T$ (Definition 4.13). We prove that $T_{\mathrm{tt}}$ is finitary with respect to $(I, \prec)$ by a well-founded induction on the order. Given any $i \in I$, with

$$R_i = (\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow j),$$

let $\Theta = [\mathsf{M}_1^{\mathcal{B}_1} {:} \lfloor \mathcal{B}_1 \rfloor, \ldots, \mathsf{M}_n^{\mathcal{B}_n} {:} \lfloor \mathcal{B}_n \rfloor]$. We verifty that $(R_i)_{\mathrm{tt}} = (\Theta \Longrightarrow \lfloor j \rfloor)$ is finitary in $T' = ((R_j)_{j \prec i})_{\mathrm{tt}}$ as follows:

- $\vdash_{T'} \Theta\ \mathsf{mctx}$ holds by induction on $k = 1, \ldots, n$: assuming $\vdash_{T'} \Theta_{(k)}\ \mathsf{mctx}$ has been established, apply (2) to a cf-derivation of $\vdash_{(R_i)_{j \prec i}} \mathcal{B}_k$ and the suitable context $\Theta_{(k)}; [\,]$.
- $\vdash_{T'} \lfloor j \rfloor$ holds by application of (2) to a cf-derivation of $\vdash_{(R_i)_{j \prec i}} j$ and the suitable context $\Theta; [\,]$.

We next address statement (2), which we prove by structural induction on the derivation of $\vdash_T \mathcal{J}$.

*Case* CF- VAR: A cf-derivation ending with the variable rule

$$\frac{}{\vdash_T \mathsf{a}^A : A}$$

is translated to an application of TT- VAR

$$\frac{\mathsf{a}^A \in |\Gamma|}{\Theta; \Gamma \vdash_{T_{\mathrm{tt}}} \mathsf{a}^A : \lfloor A \rfloor}$$

By suitability of $\Gamma$ the side-condition $\mathsf{a}^A \in |\Gamma|$ is satisfied, and $\Gamma(\mathsf{a}^A) = \lfloor A \rfloor$.

*Case* CF- META: Consider a cf-derivation ending in

$$\frac{\vdash_T t_i : A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] \quad \text{for } i = 1, \ldots, n \qquad \vdash_T \mathcal{b}[\vec{t}/\vec{x}]}{\vdash (\mathcal{b}[\vec{t}/\vec{x}])\boxed{\mathsf{M}^{\mathcal{B}}(\vec{t})}}$$

Because erasure commutes with substitution we have

$$\begin{aligned} \lfloor A_i[\vec{t}_{(i)}/\vec{x}_{(i)}] \rfloor &= \lfloor A_i \rfloor[\lfloor \vec{t}_{(i)} \rfloor/\vec{x}_{(i)}], \\ \lfloor \mathcal{b}[\vec{t}/\vec{x}] \rfloor &= \lfloor \mathcal{b} \rfloor[\lfloor \vec{t} \rfloor/\vec{x}], \\ \lfloor (\mathcal{b}\boxed{\mathsf{M}^{\{\vec{x}:\vec{A}\}\mathcal{b}}(\vec{x})})[\vec{t}/\vec{x}] \rfloor &= (\lfloor \mathcal{b} \rfloor\boxed{\lfloor \mathsf{M}^{\{\vec{x}:\vec{A}\}\mathcal{b}}(\vec{x}) \rfloor})[\lfloor \vec{t} \rfloor/\vec{x}]. \end{aligned}$$

Applying TT- META to the translation of the premises obtained by the induction hypothesis thus yields the desired result. Suitability of $\Theta; \Gamma$ is ensured because all premises are recorded in the conclusion.

*Cases* CF- META- CONGR- TY *and* CF- META- CONGR- TM: We spell out the translation of the latter rule, where $\mathcal{B} = \{x_1{:}A_1\} \cdots \{x_m{:}A_m\}\ \square : B$:

$$\frac{\begin{array}{ll} \vdash s_k : A_k[\vec{s}_{(k)}/\vec{x}_{(k)}] & \text{for } k = 1, \ldots, m \\ \vdash t_k : A_k[\vec{t}_{(k)}/\vec{x}_{(k)}] & \text{for } k = 1, \ldots, m \\ \lfloor t_k \rfloor = \lfloor t'_k \rfloor & \text{for } k = 1, \ldots, m \\ \vdash s_k \equiv t'_k : A[\vec{s}_{(k)}/\vec{x}_{(k)}] \text{ by } \alpha_k & \text{for } k = 1, \ldots, m \\ \vdash v : B[\vec{s}/\vec{x}] \qquad \lfloor \mathsf{M}^{\mathcal{B}}(\vec{t}) \rfloor = \lfloor v \rfloor & \beta \text{ suitable} \end{array}}{\vdash \mathsf{M}^{\mathcal{B}}(\vec{s}) \equiv v : B[\vec{s}/\vec{x}] \text{ by } \beta} \tag{A42}$$

The context $\Theta; \Gamma$ is suitable for the premises because $\beta$ is suitable. We apply TT- META- - CONGR as follows:

$$\frac{\begin{array}{ll} \Theta; \Gamma \vdash \lfloor s_k \rfloor : \lfloor A_k \rfloor[\lfloor \vec{s} \rfloor_{(k)}/\vec{x}_{(k)}] & \text{for } k = 1, \ldots, m \\ \Theta; \Gamma \vdash \lfloor t_k \rfloor : \lfloor A_k \rfloor[\lfloor \vec{t} \rfloor_{(k)}/\vec{x}_{(k)}] & \text{for } k = 1, \ldots, m \\ \Theta; \Gamma \vdash \lfloor s_k \rfloor \equiv \lfloor t_k \rfloor : \lfloor A_k \rfloor[\lfloor \vec{s} \rfloor_{(k)}/\vec{x}_{(k)}] & \text{for } k = 1, \ldots, m \\ \Theta; \Gamma \vdash \lfloor B \rfloor[\lfloor \vec{s} \rfloor/\vec{x}] \equiv \lfloor B \rfloor[\lfloor \vec{t} \rfloor/\vec{x}] & \end{array}}{\Theta; \Gamma \vdash \mathsf{M}^{\mathcal{B}}_k(\lfloor \vec{s} \rfloor) \equiv \mathsf{M}^{\mathcal{B}}_k(\lfloor \vec{t} \rfloor) : \lfloor B \rfloor[\lfloor \vec{s} \rfloor/\vec{x}]}$$

The first two rows of premises are secured by the induction hypotheses for the corresponding rows in (A42), and the premises in the third row are derivable by the side conditions in the third

row and induction hypotheses for the fourth row. The last premise follows by Theorem 3.8 applied to $\Theta; \Gamma \vdash_{T_{\mathrm{tt}}} \lfloor B \rfloor\ \mathsf{type}$, which holds because we assumed $\vdash_{T_{\mathrm{tt}}} \Theta\ \mathsf{mctx}$.

*Case* CF- ABSTR: A cf-derivation ending with an abstraction

$$\frac{\vdash_T A\ \mathsf{type} \qquad \mathsf{a}^A \notin \mathsf{fv}(\mathcal{J}) \qquad \vdash_T \mathcal{J}[\mathsf{a}^A/x]}{\vdash_T \{x{:}A\}\ \mathcal{J}}$$

is translated to a tt-derivation ending with TT- ABSTR

$$\frac{\Theta; \Gamma \vdash_{T_{\mathrm{tt}}} \lfloor A \rfloor\ \mathsf{type} \qquad \mathsf{b}^A \notin |\Gamma| \qquad \Theta; \Gamma, \mathsf{b}^A{:}\lfloor A \rfloor \vdash_{T_{\mathrm{tt}}} \lfloor \mathcal{J} \rfloor[\mathsf{b}^A/x]}{\Theta; \Gamma \vdash_{T_{\mathrm{tt}}} \{x{:}\lfloor A \rfloor\}\ \lfloor \mathcal{J} \rfloor}$$

The premises get their derivations from induction hypotheses, where $\mathsf{b}^A \notin |\Gamma|$ ensures that $\Gamma, \mathsf{b}^A{:}\lfloor A \rfloor$ is suitable for $\mathcal{J}[\mathsf{b}^A/x]$.

*Case of a specific rule*: Consider a derivation ending with an instantiation $I = \langle \mathsf{M}_1^{\mathcal{B}_1} \mapsto e_1, \ldots, \mathsf{M}_n^{\mathcal{B}^n} \mapsto e_n \rangle$ of a raw cf-rule $R = (\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}^n} \Longrightarrow \mathcal{b}\boxed{e})$:

$$\frac{\begin{array}{l}\vdash_T (I_{(i)*}\mathcal{B}_i)\boxed{e_i} \quad \text{for i = 1, \ldots, n} \\ \vdash_T I_*\mathcal{b}\end{array}}{\vdash_T I_*(\mathcal{b}\boxed{e})}$$

Let $\lfloor I \rfloor = \mathsf{M}_1^{\mathcal{B}_1} \mapsto \lfloor e_1 \rfloor, \ldots, \mathsf{M}_n^{\mathcal{B}^n} \mapsto \lfloor e_n \rfloor$. Because erasure commutes with instantiation we have

$$\lfloor (I_{(i)*}\mathcal{B}_i)\boxed{e_i} \rfloor = (\lfloor I \rfloor_{(i)*} \lfloor \mathcal{B}_i \rfloor)\boxed{\lfloor e_i \rfloor}$$

and $\lfloor I_*(\mathcal{b}\boxed{e}) \rfloor = \lfloor I \rfloor_* \lfloor \mathcal{b}\boxed{e} \rfloor$. Thus we may appeal to the induction hypotheses for the premises and conclude by $R_{\mathrm{tt}}$, so long as we remember to check that $\Theta; \Gamma$ is suitable for the premises, which it is because Definition 4.1 of raw cf-rules requires $\mathsf{mv}(\mathcal{b}\boxed{e}) = \{\!|\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}^n}|\!\}$.

*Case of a congruence rule*: Consider an application of the congruence rule associated with a cf-rule

$$R = (\mathsf{M}_1^{\mathcal{B}_1}, \ldots, \mathsf{M}_n^{\mathcal{B}_n} \Longrightarrow t : A),$$

as in Definition 4.8:

$$\frac{\begin{array}{ll}\vdash_T (I_{(i)*}\mathcal{B}_i)\boxed{f_i} & \text{for } i = 1, \ldots, n \\ \vdash_T (J_{(i)*}\mathcal{B}_i)\boxed{g_i} & \text{for } i = 1, \ldots, n \\ \lfloor g_i' \rfloor = \lfloor g_i \rfloor & \text{for object boundary } \mathcal{B}_i \\ \vdash_T (I_{(i)*}\mathcal{B}_i)\boxed{f_i \equiv g_i' \text{ by } \alpha_i} & \text{for object boundary } \mathcal{B}_i \\ \vdash_T t' : I_*A & \lfloor t' \rfloor = \lfloor J_*t \rfloor \\ & \beta \text{ suitable}\end{array}}{\vdash_T I_*t \equiv t' : I_*A \text{ by } \beta} \tag{A43}$$

The context $\Theta; \Gamma$ is suitable for the premises because $\beta$ is suitable. We apply the corresponding congruence for $R_{\text{tt}}$ (Definition 2.17):

$$\frac{\begin{array}{ll}\Theta; \Gamma \vdash_{T_{\text{tt}}} (\lfloor I \rfloor_{(i)*} \lfloor \mathcal{B}_i \rfloor) \boxed{\lfloor f_i \rfloor} & \text{for } i = 1, \ldots, n \\ \Theta; \Gamma \vdash_{T_{\text{tt}}} (\lfloor J \rfloor_{(i)*} \lfloor \mathcal{B}_i \rfloor) \boxed{\lfloor g_i \rfloor} & \text{for } i = 1, \ldots, n \\ \Theta; \Gamma \vdash_{T_{\text{tt}}} (\lfloor I \rfloor_{(i)*} \lfloor \mathcal{B}_i \rfloor) \boxed{\lfloor f_i \rfloor \equiv \lfloor g_i \rfloor} & \text{for object boundary } \mathcal{B}_i \\ \Theta; \Gamma \vdash_{T_{\text{tt}}} \lfloor I \rfloor_* \lfloor A \rfloor \equiv \lfloor J \rfloor_* \lfloor A \rfloor & \end{array}}{\Theta; \Gamma \vdash_{T_{\text{tt}}} \lfloor I \rfloor_* \lfloor t \rfloor \equiv \lfloor J \rfloor_* \lfloor t \rfloor : \lfloor I \rfloor_* \lfloor A \rfloor}$$

The first and the second row of premises are derivable by induction hypotheses for the corresponding rows in (A43), while the third row is derivable because of the side conditions on the third row and induction hypotheses for the fourth row. The last premise follows by Theorem 3.17 applied to $\Theta; \Gamma \vdash_{T_{\text{tt}}} A\ \mathsf{type}$, which in turn follows by induction hypothesis applied to a derivation of $\vdash_T A\ \mathsf{type}$ witnessing the finitary character of $R$.

*Case* CF- Conv- Tm: Consider a term conversion

$$\frac{\vdash_T t : A \qquad \vdash_T A \equiv B\ \mathsf{by}\ \alpha}{\vdash_T \kappa(t, \beta) : B}$$

The side condition $\mathsf{asm}(t, A, B, \alpha) = \mathsf{asm}(t, B, \beta)$ ensures that $\Theta; \Gamma$ is suitable for both premises, hence we may apply the induction hypotheses to the premisess and conclude by TT- Conv- Tm.

*Case* CF- Conv- EqTm: Consider an equality conversion

$$\frac{\vdash s \equiv t : A\ \mathsf{by}\ \alpha \qquad \vdash A \equiv B\ \mathsf{by}\ \beta}{\vdash \kappa(s, \gamma) \equiv \kappa(t, \delta) : B\ \mathsf{by}\ \alpha}$$

The side conditions

$$\mathsf{asm}(s, A, B, \beta) = \mathsf{asm}(s, B, \gamma) \qquad \text{and} \qquad \mathsf{asm}(t, A, B, \beta) = \mathsf{asm}(t, B, \delta)$$

ensure that $\Theta; \Gamma$ is suitable for both premises, hence we may apply the induction hypotheses to the premises and conclude by TT- Conv- EqTm. As in the preceding case all assumptions in the premises already appear in the conclusion, and suitability is preserved.

*Cases* CF- EqTy- Refl, CF- EqTy- Sym, CF- EqTy- Trans, CF- EqTm- Refl, CF- EqTm- Sym, CF- EqTm- Trans: These all proceed by application of induction hypotheses to the premises, followed by the corresponding tt-rule, where crucially we rely on recording metavariables in the assumption sets to make sure that $\Theta$ and $\Gamma$ are suitable for the premises.

Finally, we address statement (2), which is proved by structural induction on $\vdash_T \mathcal{B}$. The base cases CF- Bdry- Ty, CF- Bdry- Tm, CF- Bdry- EqTy, CF- Bdry- EqTm reduce to translation of term and type judgements, while the induction step CF- Bdry- Abstr is similar to the case CF- Abstr above. □

**Theorem 6.10** (Translation of standard tt- to cf-theories)

1. *For any standard tt-theory $T$ there exists a standard cf-theory $T'$ eligible for $T$.*
2. *For any $T$, $T'$ as above, if $\vdash_T \Theta\ \mathsf{mctx}$ then there exists an eligible labeling $\theta$ for $\Theta$ such that $\vdash_{T'} \theta(\mathsf{M})$ for every $\mathsf{M} \in |\Theta|$.*
3. *For any $T$, $T'$, $\Theta$, $\theta$ as above, if $\Theta; [\,] \vdash_T \Gamma\ \mathsf{vctx}$ then there exists an eligible labeling $\gamma$ for $\Gamma$ with respect to $\theta$ such that $\vdash_{T'} \gamma(\mathsf{a})\ \mathsf{type}$ for every $\mathsf{a} \in |\Gamma|$.*

4. *For any $T, T', \Theta, \theta, \Gamma, \gamma$ as above, if $\Theta; \Gamma \vdash_T \mathcal{B}$ then there exists an eligible cf-boundary $\mathcal{B}'$ for $\mathcal{B}$ with respect to $\theta, \gamma$ such that $\vdash_{T'} \mathcal{B}'$.*
5. *For any $T, T', \Theta, \theta, \Gamma, \gamma$, as above, if $\Theta; \Gamma \vdash_T \mathcal{J}$ then there exists an eligible cf-judgement $\mathcal{J}'$ for $\mathcal{J}$ with respect to $\theta, \gamma$ such that $\vdash_{T'} \mathcal{J}'$.*

***Proof*** We prove the above existence statements by explicit constructions, e.g., we prove (1) by constructing a specific $T'$ which meets the criteria, and similarly for the remaining parts. We proceed by simultaneous structural induction on all the parts.

*Proof of part* (1): We proceed by induction on a well-founded order $(I, \prec)$ witnessing the finitary character of $T = (R_i)_{i \in I}$. Consider any $i \in I$, with the corresponding specific rule

$$R_i = (\Theta \Longrightarrow \mathcal{b}\boxed{e}),$$

and let $T_i = (R_j)_{j \prec i}$. By induction hypothesis the tt-theory $T_i'$ eligible for $T_i$ has been constructred. Because $\vdash_{T_i} \Theta$ mctx, by (2) there is an eligible labeling $\theta = \langle \mathsf{M}_1 \mapsto \mathcal{B}_1', \ldots, \mathsf{M}_n \mapsto \mathcal{B}_n' \rangle$ for $\Theta$ such that $\vdash_{T_k'} \mathcal{B}_k'$ for each $k = 1, \ldots, n$. The empty map $\gamma = \langle\rangle$ is an eligible labeling for the empty context $[\,]$. Because $\Theta; [\,] \vdash_{T_i} \mathcal{b}$, by (4) there is an eligible cf-boundary $\mathcal{b}'$ for $\mathcal{b}$ with respect to $\theta, \gamma$ such that $\vdash_{T_i'} \mathcal{b}'$. We now are in possession of the cf-rule-boundary

$$\mathsf{M}_1^{\mathcal{B}_1'}, \ldots, \mathsf{M}_n^{\mathcal{B}_n'} \Longrightarrow \mathcal{b}' \tag{A44}$$

eligible for the tt-rule-boundary $\Theta \Longrightarrow \mathcal{b}$. Let

$$R_i' = (\mathsf{M}_1^{\mathcal{B}_1'}, \ldots, \mathsf{M}_n^{\mathcal{B}_n'} \Longrightarrow \mathcal{b}'\boxed{e'})$$

be the symbol or equality cf-rule induced by (A44), as in Definitions 4.5 and 4.6. Comparison with Definitions 2.14 and 2.16 shows that $[\![e']\!] = e$, as required.

*Proof of part* (2): We proceed by induction on the derivation of $\vdash_T \Theta$ mctx. The empty map is an eligible labeling for the empty metavariable context. If $\vdash_T \langle \Theta, \mathsf{M}{:}\mathcal{B} \rangle$ mctx then by inversion $\vdash_T \Theta$ mctx and $\Theta; [\,] \vdash_T \mathcal{B}$. By induction hypothesis there exists an eligible labeling $\theta$ for $\Theta$, and by (4) applied to $T, T', \Theta, \theta, [\,], \langle\rangle$ a cf-boundary $\mathcal{B}'$ eligible for $\mathcal{B}$ such that $\vdash_{T'} \mathcal{B}'$. The map $\theta' = \langle \theta, \mathsf{M} \mapsto \mathcal{B}' \rangle$ is eligible for $\langle \Theta, \mathsf{M}{:}\mathcal{B} \rangle$, and moreover $\vdash_{T'} \theta'(\mathsf{M}')$ for every $\mathsf{M}' \in |\theta'|$.

*Proof of part* (3) is analogous to part (2).

*Proof of part* (4): The non-abstracted boundaries reduce to instances of (5) by inversion, while the case of TT- Bdry- Abstr is analogous to the case TT- Abstr below.

*Part* (5): Let $T, T', \Theta, \theta, \Gamma, \gamma$ be as in (5) with

$$\begin{aligned}
\Theta &= [\mathsf{M}_1{:}\mathcal{B}_1, \ldots, \mathsf{M}_m{:}\mathcal{B}_p], \\
\theta &= \langle \mathsf{M}_1 \mapsto \mathcal{B}_1', \ldots, \mathsf{M}_p \mapsto \mathcal{B}_p' \rangle, \\
\Gamma &= [\mathsf{a}_1{:}A_1, \ldots, \mathsf{a}_p{:}A_r], \\
\gamma &= \langle \mathsf{a}_1 \mapsto A_1', \ldots, \mathsf{a}_r \mapsto A_r' \rangle.
\end{aligned}$$

We have the further assumption that each $\mathsf{M}_i$ has a cf-derivation $D_{\mathsf{M}_i}$ of $\vdash_{T'} \mathcal{B}_i'$, and each $\mathsf{a}_j$ a cf-derivation $D_{\mathsf{a}_j}$ of $\vdash_{T'} A_j'$ type. We proceed by structural induction on the derivation of $\Theta; \Gamma \vdash_T \mathcal{J}$. In each case we construct a cf-derivation concluding with $\vdash_{T'} \mathcal{J}'$ such that $\mathcal{J}'$ is eligible for $\mathcal{J}$.

*Case* TT- Var: Consider a tt-derivation ending with the variable rule

$$\frac{}{\Theta; \Gamma \vdash_T \mathsf{a}_j : A_j}$$

The corresponding cf-derivation is the application of CF- VAR

$$\frac{}{\vdash_{T'} \mathsf{a}_j^{A'_j} : A'_j}$$

*Case* TT- META: Consider a tt-derivation ending with the metavariable rule, where $\mathcal{B}_k = \{x_1{:}B_1\} \cdots \{x_m{:}B_m\}\ \mathcal{b}$ and $\mathcal{B}'_k = \{x_1{:}B'_1\} \cdots \{x_m{:}B'_m\}\ \mathcal{b}'$:

$$\frac{\Theta; \Gamma \vdash_T t_j : B_j[\vec{t}_{(j)}/\vec{x}_{(j)}] \quad \text{for } j = 1, \ldots, m \qquad \Theta; \Gamma \vdash_T \mathcal{b}[\vec{t}/\vec{x}]}{\Theta; \Gamma \vdash_T (\mathcal{b}[\vec{t}/\vec{x}])\boxed{\mathsf{M}_k(\vec{t})}}$$

The correspond cf-derivation ends with and application of CF- META,

$$\frac{\vdash_{T'} t'_j : B'_j[\vec{t}'_{(j)}/\vec{x}_{(j)}] \quad \text{for } j = 1, \ldots, m \qquad \vdash_{T'} \mathcal{b}'[\vec{t}'/\vec{x}]}{\vdash_{T'} \mathcal{b}'\boxed{\mathsf{M}^{\mathcal{B}'}(\vec{t}')}}$$

where the cf-terms $\vec{t}' = (t'_1, \ldots, t'_m)$ are constructed inductively as follows. Assuming we already have $\vec{t}'_{(j)}$, we apply the induction hypothesis to the $j$-th premise and obtain its eligible counterpart $\vdash_{T'} t''_j : B''_j$, so that $\lVert t''_j \rVert = t_j$ and $\lVert B''_j \rVert = B_j[\vec{t}_{(j)}/\vec{x}_{(j)}]$. It follows that $\lfloor B''_j \rfloor = \lfloor B'_j[\vec{t}'_{(j)}/\vec{x}_{(j)}] \rfloor$, therefore we may use Theorem 5.17 to modify $t''_j$ to a term $t'_j$ which fills $B'_j[\vec{t}'_{(j)}/\vec{x}_{(j)}]$.

*Case* TT- META- CONGR: We consider a tt-derivation ending with a metavariable term congruence rule, where $\mathcal{B}_k = \{x_1{:}B_1\} \cdots \{x_m{:}B_m\}\ \square : C$ and $\mathcal{B}'_k = \{x_1{:}B'_1\} \cdots \{x_m{:}B'_m\}\ \square : C'$:

$$\frac{\begin{array}{ll} \Theta; \Gamma \vdash_T s_j : B_j[\vec{s}_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \Theta; \Gamma \vdash_T t_j : B_j[\vec{t}_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \Theta; \Gamma \vdash_T s_j \equiv t_j : B_j[\vec{s}_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \Theta; \Gamma \vdash_T C[\vec{s}/\vec{x}] \equiv C[\vec{t}/\vec{x}] & \end{array}}{\Theta; \Gamma \vdash_T \mathsf{M}_k(\vec{s}) \equiv \mathsf{M}_k(\vec{t}) : C[\vec{s}/\vec{x}]} \tag{A45}$$

The corresponding cf-derivation ends with CF- META- CONGR- TM

$$\frac{\begin{array}{ll} \vdash_{T'} s'_j : B'_j[\vec{s}'_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \vdash_{T'} t'_j : B'_j[\vec{t}'_{(j)}/\vec{x}_{(j)}] & \text{for } j = 1, \ldots, m \\ \lfloor t'_k \rfloor = \lfloor t''_j \rfloor & \text{for } j = 1, \ldots, m \\ \vdash_{T'} s'_j \equiv t''_j : B'_j[\vec{s}'_{(j)}/\vec{x}_{(j)}]\ \mathsf{by}\ \alpha_j & \text{for } j = 1, \ldots, m \\ \vdash_{T'} v : C'[\vec{s}'/\vec{x}] & \lfloor \mathsf{M}^{\mathcal{B}}(\vec{t}') \rfloor = \lfloor v \rfloor \end{array}}{\vdash_{T'} \mathsf{M}^{\mathcal{B}}(\vec{s}') \equiv v : C'[\vec{s}'/\vec{x}]\ \mathsf{by}\ \beta} \tag{A46}$$

where suitable $\vec{s}'$, $\vec{t}'$, $\vec{t}''$, $\vec{\alpha}$, $v$, and $\beta$ remain to be constructed. The terms $\vec{s}'$ and $\vec{t}'$ are obtained as in the previous case, using the first two rows of premises of (A45). The induction hypotheses for the third row give us judgements, for $j = 1, \ldots, m$,

$$\vdash_{T'} s''_j \equiv t''_j : B''_j$$

such that $\lfloor B''_j \rfloor = \lfloor B_j[\vec{s}'_{(j)}/\vec{x}_{(j)}] \rfloor$. We convert the above equality along $\vdash_{T'} B''_j \equiv B_j[\vec{s}'_{(j)}/\vec{x}_{(j)}]$ to derive

$$\vdash_{T'} s'''_j \equiv t'_j : B_j[\vec{s}'_{(j)}/\vec{x}_{(j)}]$$

and since $\lfloor s'''_j \rfloor = \lfloor s'_j \rfloor$ by reflexivity and transitivity

$$\vdash_{T'} s'_j \equiv t'_j : B_j[\vec{s}'_{(j)}/\vec{x}_{(j)}].$$

It remains to construct $v$ and $\beta$. For the former, we apply CF- SUBST- EQTY to $\vdash_{T'} \{\vec{x} : \vec{B}'\}\ C'$ type to derive

$$\vdash_{T'} C'[\vec{s}'/\vec{x}] \equiv C'[\vec{t}'/\vec{x}] \text{ by } \delta$$

ands use it to convert $\vdash_{T'} \mathsf{M}_k(\vec{t}') : C'[\vec{t}'/\vec{x}]$ to $\vdash_{T'} \kappa(\mathsf{M}_k(\vec{t}'), \epsilon) : C'[\vec{s}'/\vec{x}]$ for a suitable $\epsilon$. We take $v = \kappa(\mathsf{M}_k(\vec{t}'), \epsilon)$ and the minimal suitable $\beta$.

*Case* TT- ABSTR: Consider a tt-derivation ending with an abstraction

$$\frac{\Theta; \Gamma \vdash_T A \text{ type} \qquad \mathsf{a} \notin |\Gamma| \qquad \Theta; \Gamma, \mathsf{a}{:}A \vdash_T \mathcal{J}[\mathsf{a}/x]}{\Theta; \Gamma \vdash_T \{x{:}A\}\ \mathcal{J}}$$

By induction hypothesis we obtain a derivation of $\vdash_{T'} A'$ type which is eligible for the first premise. The extended map $\langle \gamma, \mathsf{a} \mapsto A' \rangle$ is eligible for $\Gamma, \mathsf{a}{:}A$, and so by induction hypothesis we obtain a derivble $\vdash_{T'} \mathcal{J}'$ which is eligible for the second premise with respect to $(\theta, \langle \gamma, \mathsf{a} \mapsto A' \rangle)$. We form the desired abstraction by CF- ABSTR,

$$\frac{\vdash_{T'} A' \text{ type} \qquad \mathsf{a}^{A'} \notin \mathsf{fv}(\mathcal{J}') \qquad \vdash_{T'} \mathcal{J}'}{\vdash_{T'} \{x{:}A'\}\ \mathcal{J}'[x/\mathsf{a}^{A'}]}$$

*Case of a specific rule*: Consider a specific tt-rule

$$R = (\mathsf{N}_1{:}\mathcal{D}_1, \ldots, \mathsf{N}_m{:}\mathcal{D}_m \Longrightarrow j),$$

and the corresponding cf-rule

$$R' = ({\mathsf{N}_1}^{\mathcal{D}'_1}, \ldots, {\mathsf{N}_k}^{\mathcal{D}'_m} \Longrightarrow j')$$

Consider a tt-derivation ending with $I_* R$ where $I = \langle \mathsf{N}_1 \mapsto e_1, \ldots, \mathsf{N}_m \to e_m \rangle$:

$$\frac{\begin{array}{l}\Theta; \Gamma \vdash_T (I_{(j)*}\mathcal{D}_j)\boxed{e_j} \quad \text{for } j = 1, \ldots, m \\ \Theta; \Gamma \vdash_T I_* \mathcal{b}\end{array}}{\Theta; \Gamma \vdash_T I_*(\mathcal{b}\boxed{e})} \tag{A47}$$

The corresponding cf-derivation is obtained by an application of $R'$ instantiated with

$$I' = \langle {\mathsf{N}_1}^{\mathcal{D}'_1} \mapsto e'_1, \ldots, {\mathsf{N}_k}^{\mathcal{D}'_k} \mapsto e'_m \rangle,$$

which is constructed inductively as follows. Suppose $\vec{e'}_{(j)}$ have already been constructed in such a way that $[\![e'_k]\!] = e_k$ and $\vdash_{T'} (I'_{(k)*}\mathcal{D}'_k)\boxed{e'_k}$ for all $k < j$. The induction hypothesis for the $j$-th premise of (A47) yields $\vdash_{T'} \mathcal{D}''_j\boxed{e''_j}$ such that $\lfloor \mathcal{D}''_j \rfloor = \lfloor I'_{(j)}\mathcal{D}'_j \rfloor$. We apply Theorem 5.17 to modify $e''_j$ to $e'_j$ such that $\vdash_{T'} (I'_{(j)}\mathcal{D}'_j)\boxed{e'_j}$ and $[\![e'_j]\!] = \lfloor e''_j \rfloor$. Lastly, the premise $\vdash_{T'} I'_* \mathcal{b}'$ is derivable because $R'$ is finitary.

*Case of a congruence rule*: Consider a term tt-rule

$$R = (\mathsf{N}_1{:}\mathscr{D}_1, \ldots, \mathsf{N}_m{:}\mathscr{D}_m \Longrightarrow t : C),$$

and the corresponding cf-rule

$$R' = (\mathsf{N}_1^{\mathscr{D}'_1}, \ldots, \mathsf{N}_m^{\mathscr{D}'_m} \Longrightarrow t' : C'),$$

Given instantiations

$$I = \langle \mathsf{N}_1 \mapsto f_1, \ldots, \mathsf{N}_m \to f_m \rangle \quad \text{and} \quad J = \langle \mathsf{N}_1 \mapsto g_1, \ldots, \mathsf{N}_m \to g_m \rangle,$$

suppose the tt-derivation ends with the congruence rule for $R$:

$$\frac{\begin{array}{ll} \Theta; \Gamma \vdash_T (I_{(j)*}\mathscr{D}_j)\boxed{f_j} & \text{for } i = 1, \ldots, m \\ \Theta; \Gamma \vdash_T (J_{(j)*}\mathscr{D}_j)\boxed{g_j} & \text{for } i = 1, \ldots, m \\ \Theta; \Gamma \vdash_T (I_{(j)*}\mathscr{D}_j)\boxed{f_j \equiv g_i} & \text{for object boundary } \mathscr{D}_j \\ \Theta; \Gamma \vdash_T I_*C \equiv J_*C & \end{array}}{\Theta; \Gamma \vdash I_*t \equiv J_*t : I_*C} \tag{A48}$$

The corresponding cf-derivation ends with the congruence rule for $R'$,

$$\frac{\begin{array}{ll} \vdash_{T'} (I'_{(i)*}\mathscr{D}'_i)\boxed{f'_i} & \text{for } i = 1, \ldots, m \\ \vdash_{T'} (J'_{(i)*}\mathscr{D}'_i)\boxed{g'_i} & \text{for } i = 1, \ldots, m \\ \lfloor g''_i \rfloor = \lfloor g'_i \rfloor & \text{for object boundary } \mathscr{D}'_i \\ \vdash_{T'} (I'_{(i)*}\mathscr{D}'_i)\boxed{f'_i \equiv g''_i \text{ by } \alpha_i} & \text{for object boundary } \mathscr{D}'_i \\ \vdash_{T'} t'' : I'_*C' & \lfloor t'' \rfloor = \lfloor J'_*t' \rfloor \\ & \beta \text{ suitable} \end{array}}{\vdash_{T'} I'_*t' \equiv t'' : I'_*C \text{ by } \beta}$$

where

$$I' = \langle \mathsf{N}_1^{\mathscr{D}'_1} \mapsto f'_1, \ldots, \mathsf{N}_m^{\mathscr{D}'_m} \to f'_m \rangle \quad \text{and} \quad J' = \langle \mathsf{N}_1^{\mathscr{D}'_1} \mapsto g'_1, \ldots, \mathsf{N}_m^{\mathscr{D}'_m} \to g'_m \rangle.$$

It remains to determine $\vec{f}'$, $\vec{g}'$, $\vec{g}''$, and $t''$.

The terms $\vec{f}'$ and $\vec{g}'$ are constructed from the first two rows of premises of the tt-derivation in the same way as $\vec{e}'$ in the previous case. The third row of premises yields equations, which after an application of Theorem 5.17, take the form

$$\vdash_{T'} (I'_{(i)*}\mathscr{D}'_i)\boxed{f''_i \equiv g''_i \text{ by } \beta_i}.$$

As $\lfloor f'_i \rfloor = \lfloor f''_i \rfloor$, these can be rectified by reflexivity and transitivity to the desired form

$$\vdash_{T'} (I'_{(i)*}\mathscr{D}'_i)\boxed{f'_i \equiv g''_i \text{ by } \alpha_i}.$$

Finally, we construct $t''$ by converting $\vdash_{T'} J_*t' : J_*C$ along $\vdash_{T'} J'_*C' \equiv I'_*C'$ by $\gamma$, which is derived as follows. The induction hypothesis for the last premise of (A48) gives

$$\vdash_{T'} C_1 \equiv C_2$$

such that $\lfloor C_1 \rfloor = \lfloor I'_* C' \rfloor$ and $\lfloor C_2 \rfloor = \lfloor J'_* C' \rfloor$. Because $\vdash_{T'} I'_* C'$ type and $\vdash_{T'} J'_* C'$ type, as well as $\vdash_{T'} C_1$ type and $\vdash_{T'} C_2$ type by Theorem 5.5, we may adjust the above equation to

$$\vdash_{T'} I'_* C' \equiv J'_* C',$$

which is only a symmetry away from the desired one.

The case of a type specific rule is simpler and dealt with in a similar fashion.

*Cases* TT- EqTy- Refl, TT- EqTy- Sym, TT- EqTm- Refl, TT- EqTm- Sym: each of these is taken care of by applying the induction hypotheses to the premises, followed by application of the corresponding cf-rule.

*Cases* TT- EqTy- Trans and TT- EqTm- Trans: Consider a derivation ending with term transitivity

$$\frac{\Theta; \Gamma \vdash_T s \equiv t : A \qquad \Theta; \Gamma \vdash_T t \equiv u : A}{\Theta; \Gamma \vdash_T s \equiv u : A}$$

The induction hypotheses for the premises produce eligible judgements

$$\vdash_{T'} s' \equiv t' : A' \text{ by } \alpha \qquad \text{and} \qquad \vdash_{T'} t'' \equiv u'' : A'' \text{ by } \beta$$

Because $\lfloor A' \rfloor = \lfloor A'' \rfloor$ and $\lfloor t' \rfloor = \lfloor t'' \rfloor$, we may convert the second judgement to $A'$, and rectify the left-hand side, which results in

$$\vdash_{T'} t' \equiv u' : A' \text{ by } \gamma.$$

Now CF- EqTm- Trans applies. The case of transitivity of type equality similar and easier.

*Case* TT- Conv- Tm: Consider a conversion

$$\frac{\Theta; \Gamma \vdash_T t : A \qquad \Theta; \Gamma \vdash_T A \equiv B}{\Theta; \Gamma \vdash_T t : B}$$

The induction hypotheses for the premises produce eligible judgements

$$\vdash_{T'} t'' : A' \qquad \text{and} \qquad \vdash_{T'} A'' \equiv B' \text{ by } \alpha$$

Because $\lfloor A' \rfloor = \lfloor A'' \rfloor$, we obtain $A' \equiv B'$ by $\beta$, after which CF- Conv- Tm can be used to convert $\vdash_{T'} t'' : A'$ to a judgement $\vdash_{T'} t' : B'$ by $\beta$ which is eligible for the conclusion.

*Case* TT- Conv- EqTm: This case follows the same pattern as the previous one. □

## References


1. Aczel, P.: An introduction to inductive definitions. Stud. Logic Found. Math. **90**, 739–782 (1977)
2. Altenkirch, T., Kaposi, A.: Type theory in type theory using quotient inductive types. ACM SIGPLAN Notices **51**(1), 18–29 (2016)
3. Angiuli, C., Hou (Favonia), K.-B., Harper, R.: Cartesian cubical computational type theory: constructive reasoning with paths and equalities. In: Ghica, D., Jung, A. (eds.) CSL 2018 (2018). https://doi.org/10.4230/LIPIcs.CSL.2018.6
4. Annenkov, D., Capriotti, P., Kraus, N., Sattler, C.: Two-level type theory and applications. arXiv:1705.03307 (2019)
5. Bauer, A., Gilbert, G., Haselwarter, P.G., Pretnar, M., Stone, C.A.: Design and implementation of the andromeda proof assistant. In: TYPES'16 (2018). https://doi.org/10.4230/lipics.types.2016.5
6. Bauer, A., Haselwarter, P.G., Lumsdaine, P.L.: A general definition of dependent type theories. arXiv:2009.05539 (2020)

7. Bauer, A., Petković Komel, A.: An extensible equality checking algorithm for dependent type theories. arXiv:2103.07397 (2021)
8. Birkedal, L., Nuyts, A., Kavvos, G.A., Gratzer, D.: Multimodal dependent type theory. In: Logical Methods in Computer Science (2021). https://doi.org/10.46298/lmcs-17(3:11)2021
9. Bocquet, R., Kaposi, A., Sattler, C.: Relative induction principles for type theories. arxiv:2102.11649 (2021)
10. Capriotti, P.: Models of type theory with strict equality. PhD thesis, University of Nottingham (2016). arxiv:1702.04912
11. Cartmell, J.W.: Generalised algebraic theories and contextual categories. PhD thesis, University of Oxford (1978)
12. Cavallo, E., Mörtberg, A., Swan, A.W.: Unifying cubical models of univalent type theory. In: CSL 2020 (2020). https://doi.org/10.4230/LIPIcs.CSL.2020.14
13. Cervesato, I., Pfenning, F.: A linear logical framework. Inf. Comput. **179**(1), 19–75 (2002). https://doi.org/10.1006/inco.2001.2951
14. Charguéraud, A.: The locally nameless representation. J. Autom. Reason. **49**, 363–408 (2012)
15. Cohen, C., Coquand, T., Huber, S., Mörtberg, A.: Cubical type theory: a constructive interpretation of the univalence axiom. arXiv:1611.02108 (2016)
16. Cousineau, D., Dowek, G.: Embedding pure type systems in the Lambda-Pi-calculus modulo. In: Della Rocca, S.R. (ed.) Typed Lambda Calculi and Applications. Lecture Notes in Computer Science, pp. 102–117 (2007). https://doi.org/10.1007/978-3-540-73228-0_9
17. de Bruijn, N.G.: Lambda calculus notation with nameless dummies, a tool for automatic formula manipulation with application to the Church-Rosser theorem. Indag. Math. **75**(5), 381–392 (1972)
18. Fiore, M., Mahmoud, O.: Functorial semantics of second-order algebraic theories. arxiv:1401.4697 (2014)
19. Geuvers, H., Krebbers, R., McKinna, J., Wiedijk, F.: Pure type systems without explicit contexts. Electron. Proc. Theoret. Comput. Sci. **34**, 53–67 (2010). https://doi.org/10.4204/EPTCS.34.6
20. Gratzer, D.: Normalization for multimodal type theory. arXiv:2106.01414 (2021)
21. Harper, R.: An equational logical framework for type theories. arXiv:2106.01484 (2021)
22. Harper, R., Pollack, R.: Type checking with universes. Theoret. Comput. Sci. **89**(1), 107–136 (1991)
23. Harper, R., Honsell, F., Plotkin, G.: A framework for defining logics. J. ACM **40**(1), 143–184 (1993)
24. Haselwarter, P.G.: Effective metatheory of type theory. PhD thesis, University of Ljubljana. https://repozitorij.uni-lj.si/IzpisGradiva.php?id=134439&lang=eng (2021)
25. Isaev, V.: Algebraic presentations of dependent type theories. arxiv:1602.08504 (2016)
26. Luo, Z.: An extended calculus of constructions. PhD thesis, University of Edinburgh (1990)
27. Martin-Löf, P.: Constructive mathematics and computer programming. In: Studies in Logic and the Foundations of Mathematics, vol. 104, pp. 153–175 (1982)
28. McKinna, J., Pollack, R.: Pure type systems formalized. In: TLCA, vol. 664 (1993)
29. Petković Komel, A.: Towards an Elaboration Theorem. HoTT/UF, Invited Talk (2021)
30. Petković Komel, A.: Meta-analysis of type theories with an application to the design of formal proofs. PhD thesis, University of Ljubljana. https://repozitorij.uni-lj.si/IzpisGradiva.php?id=134058&lang=eng (2021)
31. Pfenning, F., Schürmann, C.: System description: Twelf—a meta-logical framework for deductive systems. In: Ganzinger, H. (ed.) Automated Deduction—CADE-16. Lecture Notes in Computer Science, pp. 202–206 (1999). https://doi.org/10.1007/3-540-48660-7_14
32. Pfenning, F.: Logical frameworks. In: Robinson, J.A., Voronkov, A. (eds.) Handbook of Automated Reasoning, vol. 2, pp. 1063–1147 (2001)
33. Pientka, B., Dunfield, J.: Beluga: a framework for programming and reasoning with deductive systems (system description). In: International Joint Conference on Automated Reasoning, pp. 15–21 (2010)
34. Schreiber, U., Shulman, M.: Quantum gauge field theory in cohesive Homotopy type theory. Electron. Proc. Theoret. Comput. Sci. **158**, 109–126 (2014). https://doi.org/10.4204/EPTCS.158.8
35. Streicher, T.: Semantics of Type Theory. Progress in Theoretical Computer Science. Birkhaauser, Basel (1991)
36. Tarski, A.: A lattice-theoretical fixpoint theorem and its applications. Pac. J. Math. **5**(2), 285–309 (1955)
37. Troelstra, A.S., Schwichtenberg, H.: Basic Proof Theory, vol. 43, 2nd edn. Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, Cambridge (2000)
38. Uemura, T.: A general framework for the semantics of type theory. arXiv:1904.04097 (2019)
39. Voevodsky, V.: HTS—a simple type system with two identity types (2013)
40. Watkins, K., Cervesato, I., Pfenning, F., Walker, D.: A Concurrent Logical Framework I: Judgments and Properties. Technical report, Carnegie Mellon University (2003)