Thesis for the Degree of Doctor of Philosophy

Cubical Interpretations of Type Theory

Simon Huber

UNIVERSITY OF GOTHENBURG

Department of Computer Science and Engineering
University of Gothenburg
Gothenburg, Sweden 2016

Cubical Interpretations of Type Theory

Simon Huber

Technical Report 134D
ISBN 978-91-629-0007-6 (Print)
ISBN 978-91-629-0008-3 (PDF)
Department of Computer Science and Engineering
Programming Logic Research Group

Department of Computer Science and Engineering
University of Gothenburg
SE-412 96 Göteborg
Sweden
Telephone +46 (0)31 772 1000

Printed at Chalmers Reproservice
Göteborg, Sweden 2016

Abstract

The interpretation of types in intensional Martin-Löf type theory as spaces and their equalities as paths leads to a surprising new view on the identity type: not only are higher-dimensional equalities explained as homotopies, this view also is compatible with Voevodsky's univalence axiom which explains equality for type-theoretic universes as homotopy equivalences, and formally allows to identify isomorphic structures, a principle often informally used despite its incompatibility with set theory.

While this interpretation in homotopy theory as well as the univalence axiom can be justified using a model of type theory in Kan simplicial sets, this model can, however, not be used to explain univalence computationally due to its inherent use of classical logic. To preserve computational properties of type theory it is crucial to give a computational interpretation of the added constants. This thesis is concerned with understanding these new developments from a computational point of view.

In the first part of this thesis we present a model of dependent type theory with dependent products, dependent sums, a universe, and identity types, based on *cubical sets*. The novelty of this model is that it is formulated in a constructive metatheory.

In the second part we give a refined version of the model based on a variation of cubical sets which also models Voevodsky's univalence axiom. Inspired by this model, we formulate a *cubical type theory* as an extension of Martin-Löf type theory where one can directly argue about n-dimensional cubes (points, lines, squares, cubes, etc.). This enables new ways to reason about identity types. For example, function extensionality and the univalence axiom become directly provable in the system. We prove canonicity for this cubical type theory, that is, any closed term of type the natural numbers is judgmentally equal to a numeral. This is achieved by devising an operational semantics and adapting Tait's computability method to a presheaf-like setting.

The present thesis is based on the following publications:

1. Marc Bezem, Thierry Coquand, and Simon Huber, *A model of type theory in cubical sets*, 19th International Conference on Types for Proofs and Programs (TYPES 2013) (Dagstuhl, Germany) (Ralph Matthes and Aleksy Schubert, eds.), Leibniz International Proceedings in Informatics (LIPIcs), vol. 26, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2014, pp. 107–128.

2. Simon Huber, *A model of type theory in cubical sets*, Licentiate thesis, University of Gothenburg, 2015.

3. Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg, *Cubical type theory: a constructive interpretation of the univalence axiom*, to appear in the post-proceedings of the 21st International Conference on Types for Proofs and Programs (TYPES 2015).

4. Simon Huber, *Canonicity for cubical type theory*, arXiv:1607.04156v1 [cs.LO], 2016.

Acknowledgments

First and foremost I want to thank my supervisor Thierry Coquand for his continuous guidance and support, and for our countless discussions which taught me so much and which not infrequently started as innocent questions next to the coffee machine and ended hours later with a densely filled whiteboard. I am also grateful to Nils Anders Danielsson for being my co-supervisor and for helping me out during the time Thierry was in Princeton.

My gratitude to Andy Pitts for accepting the role of the faculty opponent, and to the members of my grading committee (and the backup!): Bob Harper, Milly Maietti, Dave Sands, and Jan Smith.

I want to thank all my former and current office mates Víctor López Juan, Bassel Mannaa, Guilhem Moulin, Anders Mörtberg, Fabian Ruch, and Andrea Vezzosi not only for all the discussions at the coffee machine, in the sauna, and wherever, but also for being great friends. I am also grateful to the members of the Programming Logic group, and especially thanks to Peter Dybjer for sharing his knowledge on the history of type theory and for our discussions on higher inductive types which made the day we were stuck at several airports pass very quickly. I am also grateful to all my other colleagues and friends at the department for providing such a nice and friendly environment!

I also want to thank Helmut Schwichtenberg, one of my mentors in Munich, who helped me to find my PhD position in Gothenburg, and my former colleagues and friends at the LMU.

Over the years I have also benefited a great deal from discussions with numerous people: Thorsten Altenkirch, Carlo Angiuli, Jean-Philippe Bernardy, Marc Bezem, Guillaume Brunerie, Cyril Cohen, Martín Escardó, Kuen-Bang Hou (Favonia), Marcelo Fiore, Fredrik Nordvall Forsberg, Ambrus Kaposi, Peter LeFanu Lumsdaine, Dan Licata, Andrew Polonsky, Christian Sattler, Peter Schuster, Bas Spitters, Andrew Swan, Chuangjie Xu, the members of the "cubical seminar" at the IHP, and everyone else I forgot to mention in this long list—thanks!

For their constant support I want to thank my family and my friends, and especially my nephews for their cheerfulness. Many thanks also to Sofie Kindahl for always being there for me!

Contents

Introduction

The recently discovered connection between Martin-Löf's intuitionistic theory of types and the seemingly unrelated field of homotopy theory sheds new light on the treatment of equality in type theory. In particular, Voevodsky's univalence axiom explains the equality for type-theoretic universes and has many useful consequences for the encoding of mathematics in type theory. The present thesis investigates the problem of giving a *computational justification* of this new development.

We will now give a brief overview of intuitionistic type theory and its notions of equality, the relation of type theory with homotopy theory and the univalence axiom, continuing with discussing the problem of justifying this axiom computationally, and explain how this problem is addressed in the remainder of this thesis.

1 Intuitionistic Type Theory

The *intuitionistic theory of types* [62, 59, 60, 61] was devised by Martin-Löf as a formal logical system and philosophical foundation for constructive mathematics. It is based on the propositions-as-types interpretation, which identifies a proposition with the type of its proofs; intuitionistic type theory extends this interpretation—also referred to as *Curry-Howard correspondence*—to predicate logic using the crucial ingredient of *dependent types*, that is, types which may depend on terms. It internalizes the Brouwer-Heyting-Kolmogorov explanation of the logical connectives.

A distinct feature of Martin-Löf's theory of types is that it may as well be viewed as a typed functional programming language, much in the style of Haskell or the ML family of languages and is thus also of particular interest from the perspective of computer science. Viewed as a programming language it has two important aspects: (1) All programs terminate. (2) It has a very expressive type system which allows to precisely state a formal specification (task) of a program; thus synthesizing a program and proving its correctness can be done at the same time and in the same framework.

Such a formal system based on dependent types was also independently proposed by de Bruijn [32] in his pioneering work on the proof checker Automath beginning in the late sixties. Also today, intuitionistic type theory is

the basis of many of the current proof assistants such as Agda, Nuprl, and Coq (the latter based on the related Calculus of Constructions by Coquand and Huet [30]). As such it has been used successfully in mechanizing large-scale proofs like the Four Color Theorem [41] and the Feit-Thompson Theorem [42] from group theory.

2 Equality

One particularly intricate concept of intuitionistic type theories is *equality*. Indeed the treatment of equality was one of the important components which changed during the various different formulations of intuitionistic type theory. There are two key notions of equality in type theory.

First, each of the different formulations of type theory comes with some form of *judgmental equality*. In the modern formulations of type theory this is presented as the two judgments

$$A = B,$$
$$a = b : A,$$

reading as A and B are equal types, and a and b are equal objects of type A, respectively. These are additionally to the other judgments of type theory: A is a type, and $a : A$, that is, a is of type A. The equality judgments enter the typing judgments by the crucial type conversion rule: from $a : A$ and $A = B$ we can derive $a : B$. This rule is required to get the very dynamics of dependent types to work, see, for example, the motivation given in [59, p. 86]. This judgmental equality is sometimes given as a *definitional equality*, that is, equality up to the unfolding of definitions;[1] often this notion is deeply connected with the computational aspects of type theory.

Judgmental equality is however a judgment of type theory and as such does not appear as a type itself. This role is taken by the second notion of equality, the *identity type*: given a type A and a and b of type A, the identity type $\mathsf{Id}_A(a, b)$ serves as type of proofs witnessing that a and b are identical. By the propositions-as-types interpretation, the identity type is thus often referred to as *propositional equality*. It is also the prime example of a dependent type.

Martin-Löf formulated two main variations of the identity type, the *intensional* identity type from [59], and the *"extensional"* identity type from [60, 61]. In the latter one has the so-called equality reflection rule: from a proof of $\mathsf{Id}_A(a, b)$ one can deduce that a and b are judgmentally equal, that is, $a = b : A$. This equality reflection rule, however, destroys the decidability of judgmental equality, and henceforth also the decidability of type checking—resulting in a formal system where for a given syntactic entity it is neither decidable whether this syntactical entity is a proposition nor whether it is a proof of a given proposition. The intensional identity type on the other hand retains decidability of

[1] The notion of definitional equality also appears in the work of de Bruijn [32] where it coincides with judgmental equality.

type checking by dropping the equality reflection rule but still keeping its introduction and elimination rule.[2] The introduction rule (also present for the extensional identity type) witnesses reflexivity, that is,

$$\frac{a : A}{\mathsf{r}\, a : \mathsf{Id}_A(a, a)}$$

and the elimination rule (also called the J-rule) states[3]

$$\frac{\begin{array}{c} x : A, y : A, z : \mathsf{Id}_A(x, y) \vdash C(x, y, z) \\ x : A \vdash d(x) : C(x, x, \mathsf{r}\, x) \qquad a : A \qquad b : A \qquad p : \mathsf{Id}_A(a, b) \end{array}}{\mathsf{J}(x.d(x), a, b, p) : C(a, b, p)}$$

with the judgmental equality (tacitly assuming well-typedness)

$$\mathsf{J}(x.d(x), a, a, \mathsf{r}\, a) = d(a) : C(a, a, \mathsf{r}\, a). \tag{\#}$$

The elimination rule adapts the concept of equality as the least reflexive relation to dependent types—this purely formal adaption has, as we shall see later, far-reaching and unexpected connections to homotopy theory. From the elimination rule we can also derive Leibniz's principle of indiscernability of identicals.

The intensional identity type, while retaining decidability of type checking, is, however, per se often too restrictive in practice to encode mathematics since it lacks extensional concepts, most notably function extensionality which states that two functions are propositionally equal whenever they agree on all arguments. The intensional identity type only identifies (closed) functions if they are judgmentally equal; more generally, using a normalization argument one can show [59] that in the empty context, $\mathsf{Id}_A(a, b)$ is inhabited iff $a = b : A$. The "extensional" identity type on the other hand allows to derive function extensionality using the congruence and η-rule together with equality reflection.

Another extensionality concept is *quotient types*, that is, the possibility to redefine equality by a given relation. This is neither directly available for intensional nor "extensional" identity types.

One way to obtain function extensionality is to simply add it as an axiom. The *canonicity* property for the type of natural numbers N states that any closed term of type N is judgmentally equal to a numeral (that is, a term only built from zero and successors). This does *not* hold in the presence of function extensionality stated as an axiom. This canonicity property is in particular desirable from the perspective of intuitionistic type theory as a programming language: one should be able to evaluate/run a (closed) program of type N to

[2] In the absence of the equality reflection rule it often makes sense to use judgmental and definitional equality as synonyms (although to refer to η-equality as an equality by "definition" might also be questioned).

[3] We write $\Gamma \vdash \mathcal{J}$ for *hypotetical judgments* where Γ is a context of the form $x_1 : A_1, \ldots, x_n : A_n$, that is, each A_i is a type in the context $x_1 : A_1, \ldots, x_{i-1} : A_{i-1}$.

obtain a numeral and moreover the theory should know about this numeral (namely via judgmental equality).

A possible remedy to the lack of function extensionality is to exploit the *setoid interpretation* of type theory by Hofmann [46] and to work instead with setoids, that is, a type equipped with an equivalence relation (to serve as propositional equality relation). Functions, under this interpretation, are required to preserve the respective equivalence relation and thus are extensional. This is inspired by Bishop's notion of set in which "a *set* is defined by describing what must be done to construct an *element* of the set, and what must be done to show that two elements of the set are *equal*" [18, p. 67]. Hofmann's interpretation, however, does not satisfy all judgmental equalities of type theory and its use turns out to be rather cumbersome in practice, especially in the presence of dependent types. A system incorporating ideas from the setoid interpretation and reconciling some of the extensional concepts with intensional type theory is observational type theory [6]; this system extends type theory by a type of propositions with proof irrelevance.

The rich structure of the identity type stems from the fact that iterating the identity type gives rise to *higher-dimensional equalities*: given a and b of type A, we can form $\mathsf{Id}_A(a,b)$; given p and q of type $\mathsf{Id}_A(a,b)$, we can form the type $\mathsf{Id}_{\mathsf{Id}_A(a,b)}(p,q)$; given further α and β of type $\mathsf{Id}_{\mathsf{Id}_A(a,b)}(p,q)$, we can form

$$\mathsf{Id}_{\mathsf{Id}_{\mathsf{Id}_A(a,b)}(p,q)}(\alpha,\beta);$$

and so on. The natural question now arises: what structure do these higher identity types form? In particular, can one prove $\mathsf{Id}_{\mathsf{Id}_A(a,b)}(p,q)$ for p and q of type $\mathsf{Id}_A(a,b)$, that is, is there essentially only "one way" to prove a propositional equality $\mathsf{Id}_A(a,b)$? The latter principle is often referred to as *uniqueness of identity proofs* (UIP). In extensional type theory UIP is provable from the reflection rule and thus this hierarchy collapses. Although it is known by a theorem of Hedberg [45] that any type with *decidable equality*[4] satisfies UIP, the general answer to this question is however negative as shown by the pioneering work of Hofmann and Streicher [49, 46]. They devise a model of intensional type theory where a (closed) type A is interpreted by a *groupoid*[5] G and the closed terms of A are interpreted as the objects in G.[6] The arrows $a \to b$ for a and b objects of G stand for the witnesses that a and b are propositionally equal. UIP is then refuted by specifying a groupoid with two distinct parallel arrows $a \to b$. The fact that G is required to be a groupoid stems from the fact that one can define—internally in type theory—operations corresponding to the groupoid operations: the introduction rule $\mathsf{r}\,a : \mathsf{Id}_A(a,a)$ corresponds to the identity map, transitivity

$$\_ \circ \_ : \mathsf{Id}_A(b,c) \to \mathsf{Id}_A(a,b) \to \mathsf{Id}_A(a,c)$$

[4] That is, a type A such that $\mathsf{Id}_A(x,y) + \neg\,\mathsf{Id}_A(x,y)$ for all x and y of type A.

[5] A category where each arrow is invertible.

[6] A model for dependent types based on groupoids also has been studied earlier by Lamarche [55] without considering identity types.

corresponds to composition, and symmetry

$$\_^{-1} : \mathsf{Id}_A(a,b) \to \mathsf{Id}_A(b,a)$$

corresponds to taking the inverse. These operations satisfy the expected groupoid equations, but in general only *up to propositional equality*; for example, $(p \circ q) \circ r$ (for appropriately typed p, q, r) is not definitional equal to $p \circ (q \circ r)$, but we can define a *higher* equality between equalities

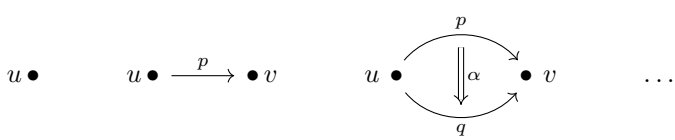
$$\alpha_{p,q,r} : \mathsf{Id}_{\mathsf{Id}_A(a,d)}((p \circ q) \circ r, p \circ (q \circ r)).$$

The groupoid interpretation validates function extensionality and an extensionality principle for universes stating that propositional equality for universes is isomorphic to having an isomorphism. (A type-theoretic universe is a type of types.) It should also be noted that the groupoid model is given in a *constructive* metatheory; but it is not clear whether it can be directly formalized within intensional type theory (that is, type theory with the intensional identity types).

3 Homotopy Theory and Type Theory

Already the groupoid interpretation suggests that a type in intensional type theory should be thought of more than merely a "set". Instead, a type should be thought of as a topological space—but up to *homotopy*. Around 2006, Awodey and Warren [9] and Garner [38] discovered connections between homotopy theory and type theory in the context of Quillen model categories, in particular, between the J-rule and the abstract lifting conditions of those model categories. Moreover, Streicher [76] and independently Voevodsky [82, 54] built a model of type theory using Kan simplicial sets (the latter model also supporting type-theoretic universes, that is, types of types).

One crucial point in this analogy between types and spaces is what under this view the interpretation of the identity type is: while $u : A$ should be thought of as a point in the space A, a term $p : \mathsf{Id}_A(u,v)$ should be thought of as a *path* between u and v, that is, a continuous map $p\colon \mathbb{I} \to A$ (where $\mathbb{I} = [0,1]$ is the interval) such that $p(0) = u$ and $p(1) = v$. Higher equalities $\alpha : \mathsf{Id}_{\mathsf{Id}_A(u,v)}(p,q)$ correspond to homotopies (or 2-dimensional paths) $\alpha\colon \mathbb{I} \to A^{\mathbb{I}} \cong \mathbb{I} \times \mathbb{I} \to A$ between the paths $p, q\colon \mathbb{I} \to A$ with $\alpha|_{\{0\}\times\mathbb{I}} = p$ and $\alpha|_{\{1\}\times\mathbb{I}} = q$; etc.



All the higher homotopies on a space A organize into a structure called the *fundamental ∞-groupoid* of A, the prime example of a so-called *(weak) ∞-groupoid*. These higher groupoids are closely connected to homotopy theory,

as proposed by Grothendieck [44]. It was shown that a type (in intensional type theory) and its higher-dimensional equalities give rise to such an ∞-groupoid [58, 57, 80].

Another groundbreaking insight by Voevodsky besides the interpretation of type theory using Kan simplicial sets was that this interpretation satisfies an additional axiom, the so-called *univalence axiom* [87]. This axiom explains the identity type for type-theoretic universes. Such a type-theoretic universe U is a type of types: in the presentation of universes à la Russell a term $A : \mathsf{U}$ is also a type A; such universes are usually closed under type formers such as dependent functions, dependent sums, natural numbers, and identity types. The univalence axiom states, loosely speaking, that *equivalent*[7] types (as elements of U) are equal, where "equal" refers to the identity type of U. This formalizes the everyday habit by mathematicians to identify isomorphic structures, although this is clearly not valid in set theory: $0 \in \{0\}$ but $0 \notin \{1\}$ even though $\{0\} \cong \{1\}$. The univalence axiom can also be seen as an extensionality principle: it is a natural generalization of the concept in simple type theory formulated by Church [23] that two propositions are equal given that they are logically equivalent. It also generalizes the notion of universe extensionality discovered by Hofmann and Streicher in their groupoid interpretation. Univalence is further incompatible with the UIP principle since, for example, negation gives a non-trivial automorphism on N_2 (the booleans), which induces a non-trivial proof of equality by univalence.

Voevodsky also suggested a new foundation of mathematics, called *univalent foundations* [84, 83, 85]. In this foundation the domain of discourse is not given by sets but rather, more general, by their "higher-dimensional analogues" of ∞-groupoids. Such ∞-groupoids are stratified by the notion of n-groupoids which correspond to so-called homotopy n-types (spaces, up to homotopy equivalences, where the i-th homotopy group vanishes for all $i > n$). It has been argued in [84] that categories are not higher-level analogues of sets, but rather this role is taken by groupoids. From this perspective a category is a "groupoid" equipped with additional structure (sets of morphisms with identities and compositions satisfying certain equations), being the next level analogue of a partially ordered set: a set equipped with the additional structure of a binary relation which is reflexive and transitive.

Voevodsky realized that the language of intensional Martin-Löf type theory is suitable to express this view of mathematics.[8] One of his crucial observations was that one can express the above stratification *inside* type theory: the important hierarchy of *homotopy levels*, or *h-levels* is given as follows. Call a type A *contractible*, or of h-level 0, if we have

$$(\Sigma a : A)(\Pi x : A)\ \mathsf{Id}_A(a, x).$$

[7] That is types, having an equivalence between them; for the sake of this introduction the technical definition of equivalence is not so important; let us just mention that being an equivalence is logically equivalent to being an isomorphism, that is, having a (pointwise) inverse.

[8] See in particular the formalization [88, 89] developed in the type theory of Coq with univalence added as an axiom.

A type A is of h-level $n+1$ if for all $x, y : A$ the type $\mathsf{Id}_A(x, y)$ is of h-level n. It is a theorem that a type is of h-level 1 iff it has at most one element; we call those *h-proposition*[9]. For a type to be of h-level 2 is then the same to have UIP for this type (those are called *h-sets*). Types with h-level 3 are called *h-groupoids*. The univalence axiom and the J-rule imply that this hierarchy is "well behaved", in particular it is *cumulative*. It is also closed under dependent function types, that is, $(\Pi x : A)B(x)$ is of h-level n whenever $B(x)$ is a family of types of h-level n; this uses that in fact function extensionality is a consequence of univalence. Moreover, one can show that the type of all types of h-level n is of h-level $n+1$; in particular, the type of h-sets is an h-groupoid.

Also the notion of equivalence referred to in the formulation of the univalence axiom, formulated of course inside type theory, was a major contribution by Voevodsky. It generalizes the notion of logical equivalence, isomorphism/bijection between sets, categorical equivalence, etc., if specialized to objects with according h-levels. Moreover, the univalence axiom implies that isomorphic structures are equal, for example, two groups (formulated as h-sets with units and binary operations satisfying the usual laws) which are isomorphic are equal.

Because of the intricate connections between homotopy theory and type theory this branch of mathematics is often called *homotopy* or *univalent type theory*. The univalence axiom with many of its discussed consequences—among others—allows for a better encoding of mathematics in dependent type theory à la Martin-Löf, as is particularly shown in the book [79]. Another aspect, developed, for example, *ibid.* and in Brunerie's recent thesis [20], is that type theory can also be used to develop algebraic topology where all notions are invariant under homotopy equivalence.

4 Computational Hurdles

Similar as for the function extensionality axiom, simply postulating the univalence axiom as a constant added to intensional Martin-Löf type theory destroys the good computational behavior of type theory, making it necessary to explain univalence computationally. One possible attempt to do so is to build a model of this axiom in type theory itself or at least in a constructive metatheory. Such a computational interpretation could then be obtained through semantics, for example, by evaluating a term of type N (the natural numbers) in the model. A computational explanation obtained in this way may however have the defect that *a priori* it is not clear that if we would have another such explanation via evaluation that we would get the same numeral. This would however follow from a conjecture by Voevodsky [86]:

Conjecture. *There is a terminating algorithm that for any $u : \mathsf{N}$ which is closed except that it may use the univalence axiom returns a closed numeral*

[9] This notion should not be confused with the notion of proposition as we used it earlier in the proposition-as-types interpretation. However, often the suffix "h-" is also dropped.

$n : \mathsf{N}$ *not using the univalence axiom and a proof that* $\mathsf{Id}_{\mathsf{N}}(u, n)$ *(which may use the univalence axiom).*

As mentioned above, Voevodsky devised a model of the univalence axiom in simplicial sets and thus ensuring its consistency. Roughly speaking, simplicial sets are a combinatorial representation of spaces; a simplicial set is given by specifying its n-simplicies: more formally, a simplicial set is a presheaf on the simplicial category Δ whose objects are the non-empty finite linear ordered sets $[n] = \{0, 1, \ldots, n\}$, $n \geq 0$, and whose morphisms are (non-strictly) order preserving maps. Among those there are the Kan simplicial sets, that is, simplicial sets satisfying Kan's extension condition: any horn–simplices missing their interior and one face–can be filled. A way to look at simplicial sets is as a generalization of (reflexive) relations: points are related if they are connected by an edge (1-simplex); from this perspective, Kan simplicial sets generalize sets equipped with an equivalence relation, that is, setoids. Kan simplicial sets are also one way to make the notion of ∞-groupoid precise.

In Voevodsky's model of the univalence axiom closed types are interpreted by Kan simplicial sets (and general types by Kan fibrations). This model is however formulated in classical ZFC set theory and the theory of (Kan) simplicial sets relies essentially on *classical* reasoning; as such, it can *not* directly be used to explain univalence computationally. One aspect where classical logic is used essentially arises in the general theory of simplicial sets: being a degenerate simplex is not a decidable property. Reasoning by cases on whether a simplex is degenerate or not is used, for example, when showing that for a Kan fibration a path in the base induces an equivalence of the fibers. Another example where this is used is to show that B^A is a Kan simplicial set whenever B is. Both of these classical theorems are not provable intuitionistically, as shown using Kripke models in [14] and [16], respectively. The latter result was also strengthened in Parmann's thesis [67] to include Kan simplicial sets where the Kan fillers are explicitly given as functions.

5 This Thesis

This thesis is concerned with giving a *constructive* justification of the univalence axiom. In a first attempt [11], not included in this thesis, we were using a model based on Kan *semi*-simplicial sets[10]. This approach is however very involved and does not model various laws for substitutions. Instead, this thesis addresses the question of giving a constructive justification of the univalence axiom by concentrating on methods involving *cubical sets*. Cubical sets were used to give the first *combinatorial* definition of homotopy groups by Kan [53].

In this thesis we will use two variations of cubical sets, and both of the variants view cubes as *formal* representations of cubes seen as continuous maps $u \colon \mathbb{I}^J \to X$ (with $\mathbb{I} = [0, 1]$) where J is a finite set of so-called *names* $i_1, \ldots, i_n$,

[10] Semi-simplicial sets are defined like simplicial sets, but instead of the simplicial category Δ one takes the category with the same objects but *strictly* monotone maps instead; this results in the fact that semi-simplicial sets are not equipped with degeneracy operations.

instead of the more common $u\colon \mathbb{I}^n \to X$. We want to view such a cube as "value depending on the names" $i_1, \ldots, i_n$. On such a value we can perform basic operations which correspond to *substitutions* (or *reparametrizations*) of the variables $i_1, \ldots, i_n$. The two variants of cubical sets we consider differ in the substitutions allowed; both of them include: the face operation, that is, setting a variable to 0 or 1, so, for example, $u(i_k = 0)$; adding a variable dependency for a fresh name j, so considering u from above as a value depending on $i_1, \ldots, i_n, j$, constant in the j-direction—this corresponds to the degeneracy operation; and, renaming a variable i_k into a fresh name j, $u(i_k = j)$. These formulations bear close resemblance to the theory of nominal sets [70, 69, 71]. There are various further variations of cubical sets used in the literature (see, for example, [43]).

Structure of this Thesis

This thesis is structured into two parts which can be read independently. Part I is, apart from minor corrections and the omission of abstract and acknowledgments, a copy of my licentiate thesis [50], which in turn is based in parts on [15]. As such the introduction to Part I has a slight overlap with the present introduction. Part II comprises two chapters, each based on a paper adapted to match the present layout: Chapter 6 is based on [26] and Chapter 7 is based on the preprint [51]. We will now give a short overview of each of the three components of this thesis.

5.1 A Model of Type Theory in Cubical Sets

In Part I we begin by exploring a model of type theory not based on simplicial sets but rather on a variant of cubical sets. Similar to simplicial sets, cubical sets are formally defined as presheaves, but on a category of cubes whose objects are finite sets of names $I = \{i_1, \ldots, i_n\}$, $n \geq 0$, and morphisms are substitutions $J \to I$ described by set-theoretic maps $I \to J \cup \{0, 1\}$ required to be injective on the preimage of J—these formally describe the operations mentioned above.

Since any presheaf category induces a model of type theory (see, for example, [47]) we get such a model where the contexts are interpreted by cubical sets. This model however satisfies UIP and does not give us the envisaged identity types—we want that identity types should be modeled, in accordance to the above correspondence to homotopy, namely as *path spaces*. Similar to the model based on Kan simplicial sets where types are interpreted by Kan fibrations, we have to strengthen the notion of types and have to require a so-called Kan structure. This structure refines Kan's original extension condition ("Kan cubical sets") as defined in [53] which amounts to have fillers for all open boxes. We refine Kan's notion in three aspects: first, we require these fillers to be explicitly given as operations, second, we allow more general shapes to be filled (for example, a line can be extended to a square), and, third, that these operations satisfy certain *uniformity conditions* which ensure that the filling

operations commute with the name substitutions in a suitable way. The third refinement is essential to show that types are closed under dependent function spaces.

This way we obtain a model of Martin-Löf type theory supporting dependent functions and sums, identity types, and universes. The identity types are interpreted by path spaces induced by (affine) exponentials with an interval. These, however, are only *weak* identity types in the sense that they satisfy the judgmental equality (#) only propositionally. A proper identity type can be recovered following an idea of Swan [77]. This model is given in a constructive metatheory and thus gives rise to an effective method to compute.

Inspired by (a nominal version of) this model we have implemented a proof checker [25] in Haskell. This also features an experimental implementation of the univalence axiom based on the note [28].

This model has also been considered from the perspective of nominal sets by Pitts [69, 71]. Moreover, Swan [77] analyzes the model using the language of algebraic weak factorization systems [39] in the constructive set theory CZF.

5.2 Cubical Type Theory: A Constructive Interpretation of the Univalence Axiom

We begin Part II in Chapter 6 by formulating a *cubical type theory* as an extension of Martin-Löf type theory which is inspired by a refinement of the model described in Part I. This type theory allows to directly argue about n-dimensional cubes, function extensionality has a direct proof, and Voevodsky's univalence axiom is provable. One of its important ingredients is that expressions may depend on names ranging over a *formal* interval $\mathbb{I}$, that is, we allow contexts also to contain variable declarations $i : \mathbb{I}$; a term $u(i) : A$ depending on $i : \mathbb{I}$ (with, say, A a closed type) can then be considered as a *line* with endpoints $u(0) : A$ and $u(1) : A$. Such a term $u(i)$ induces a *path* $\langle i \rangle\, u$ between $u(0)$ and $u(1)$ by abstracting the variable i.

The notion of cubical sets which constitutes a model for cubical type theory, is similarly to the one in Part I but where the category of cubes is now refined by taking as morphisms $J \to I$ maps $I \to \mathsf{dM}(J)$ where $\mathsf{dM}(J)$ is the free de Morgan algebra with generators in J; this allows also for operations corresponding to connections [19] and to taking diagonals.

Each type, both in the model and type theory, comes equipped with so-called *composition operations*[11] which, roughly speaking, give only the "lid" of an open box. In the model, these operations are required to satisfy similar uniformity conditions as in the model described in Part I; on the syntax side, that is, in cubical type theory, this uniformity corresponds to the fact that these operations commute with substitutions. A significant simplification over the model in Part I is that the filling operations can be defined from composition operations using connections.

[11] In that respect $\mathbb{I}$ is not a type since it can not be equipped with such composition operations as shown by Christian Sattler (private communication).

Another important ingredient of cubical type theory and its model is the *glueing construction*. It can be used to explain the composition operation for universes and to justify the univalence axiom. The glue operation allows for a type to glue types to certain parts of another type along an equivalence.

We support the usual type formers for dependent function spaces and dependent sums, universes, data types, and path types. The path type is a weak identity type in the above sense; it is also possible to extend, both the semantics and the syntax, by a proper identity type on top of the path type also satisfying the judgmental equality (#) following Swan's trick.

We also support certain *higher inductive types* like spheres and propositional truncation. Such a higher inductive type is similar to an inductive type but one may also introduce constructors for paths. In this way, the circle can, for example, be defined with a point constructor base and a path constructor loop giving a path between base and base.

We have implemented a prototype proof assistant [27] for cubical type theory in Haskell.

Parts of this model have been generalized to other settings by Gambino and Sattler [37]. Orton and Pitts [66] investigate on how the model can be formulated in the internal language of a topos assuming an interval-like object. In [17] cubical type theory has been extended to incorporate ideas from guarded type theory. Mark Bickford has reported on an ongoing formalization of the model in the Nuprl proof assistant.

5.3 Canonicity for Cubical Type Theory

Although the consistency of cubical type theory as formulated in Chapter 6 follows from its model in cubical sets, we would also like to establish important properties like normalization and decidability of type checking. As a first step to address these problem we prove a canonicity theorem in Chapter 7: given a context I only built from name variables, that is, of the form $i_1 : \mathbb{I}, \ldots, i_k : \mathbb{I}$, $k \geq 0$, and a derivation $I \vdash u : \mathsf{N}$, then there is a $n \in \mathbb{N}$ such that $I \vdash u = \mathsf{S}^n\, 0 : \mathsf{N}$. It is moreover effective to obtain this number n. The proof is an adaption of Tait's computability method [78, 62] to a presheaf-like setting; we work with a typed and deterministic operational semantics.

6 Statement of Personal Contribution

As already mentioned above, parts of this thesis are based on collaborations with other people. My personal contribution is as follows:

Part I. This part is based on my licentiate thesis which in turn is based in parts on [15], coauthored with Marc Bezem and Thierry Coquand. In this paper (*loc cit.*), the crucial ingredient of the uniformity condition is due to Coquand; I mainly worked out the details of the semantics and in particular the details of the Kan structure for Π-types and the sketch for compositions in

universes. The write-up in the licentiate thesis is entirely by me and includes more details. The proof that the universe is a Kan cubical set is by me and not included in [15].

Part II, Chapter 6. This chapter is based on the paper [26] coauthored with Cyril Cohen, Thierry Coquand, and Anders Mörtberg, which is to appear in the post-proceedings of the 21st International Conference on Types for Proofs and Programs, TYPES 2015. The ideas in the paper came about in the collaborative effort to implement the type checker [25, 27] for [15]. The initial use of connections and the glueing operation (generalizing an operation sketched in [15]) is due to Coquand. I am the main responsible for the section about semantics. I also participated working on all the other parts of the paper. Together with Mörtberg, I discovered that the glueing operation satisfies to prove the univalence axiom. See also the notes in the Acknowledgment section of this chapter for my contributions during the development of the material.

Part II, Chapter 7. This chapter is based on the preprint [51] which is entirely my own work.

Part I

A Model of Type Theory in Cubical Sets

Introduction

Dependent type theory has been successfully used as a foundation to develop formalized mathematics and computer science. This is reflected not only by the popularity of the proof assistants Coq and Agda (among others) based on type theory, but also the formal proofs of the Four Color Theorem [41] and the Feit-Thompson (Odd Order) Theorem [42] show that non-trivial mathematics can be encoded in type theory.

There is however one especially intricate concept in dependent type theory: *equality*. The *identity type*, $\mathtt{Id}_A(u, v)$, for u and v of type A, serves as the type of proofs witnessing that u and v are *identical*. Dependent type theory also comes with a different notion of equality namely *definitional equality* which, in contrast to the identity type, is not a type (hence the identity type is often referred to as *propositional equality*), but definitional equality concerns the computational aspect of type theory, and is often presented as a judgment (and hence also referred to as *judgmental equality*). Martin-Löf formulated two variations of the identity type, the intensional identity type from [59], and the "extensional" identity type from [61]. In the latter one has the so-called *reflection rule*: if we have a proof of $\mathtt{Id}_A(u, v)$, we can deduce that u and v are definitionally equal. This equality reflection rule, however, destroys the decidability of type checking, resulting in a formal system where it is not decidable whether a given syntactic entity is in fact a proof of a given proposition. The intensional identity type on the other hand, drops this rule while still keeping its introduction and elimination rules (the latter often referred to as the J-rule), which entail the usual properties of equality, in particular Leibniz's rule of indiscernability of identicals.

While the formulation without equality reflection retains decidability of type checking, the intensional identity type per se is, however, often too restrictive to directly encode certain mathematical practices such as identifying two functions which are extensionally equal, that is, for which each argument gives equal results. The intensional identity type only identifies functions which are definitionally equal which often tends to be too restrictive in practice. One remedy to this particular problem is to simply add functional extensionality as an axiom—but this destroys the good computational properties of type theory: e.g., one can define a closed term of type N (the type of natural numbers) which is *not* definitionally equal to a numeral (i.e., a term merely built up from successors and zero). One possibility is to exploit Hofmann's [46] *setoid inter-*

pretation and work with *setoids* (a type with an equivalence relation) instead, and relativize to functions preserving this relation. This, however, turns out to be rather cumbersome in practice. A type theory incorporating ideas from the setoid interpretation and reconciling some of the extensional concepts with intensional type theory is observational type theory [6].

The rich structure of the intensional identity type stems from the fact that in type theory we can *iterate* the identity type to obtain "higher-dimensional" identity types: for p and q of type $\mathtt{Id}_A(u,v)$ we can form $\mathtt{Id}_{\mathtt{Id}_A(u,v)}(p,q)$; and for α and β of type $\mathtt{Id}_{\mathtt{Id}_A(u,v)}(p,q)$ we can furthermore form the type

$$\mathtt{Id}_{\mathtt{Id}_{\mathtt{Id}_A(u,v)}(p,q)}(\alpha,\beta);$$

etc. It is a natural question to ask what the structure of these higher identity types is. In particular, can one prove $\mathtt{Id}_{\mathtt{Id}_A(u,v)}(p,q)$ for p and q of type $\mathtt{Id}_A(u,v)$, i.e., is there essentially only "one way" to prove a propositional equality $\mathtt{Id}_A(u,v)$? The latter principle is often referred to as *uniqueness of identity proofs* (UIP). In extensional type theory UIP is provable from the reflection rule and thus this hierarchy collapses. For intensional type theory, however, UIP is not provable as shown by the pioneering work of Hofmann and Streicher [49]. They devise a model of intensional type theory where a (closed) type A is interpreted by a *groupoid*[12] G and the closed terms of A are interpreted as the objects in G. The arrows $u \to v$ for u and v objects of G stand for the witnesses that u and v are propositionally equal. UIP is then refuted by specifying a groupoid with two distinct parallel arrows $u \to v$. The fact that G is required to be a groupoid stems from the fact that one can define–internally in type theory–operations corresponding to the groupoid operations: the introduction rule $\mathtt{refl}\,u : \mathtt{Id}_A(u,u)$ corresponds to the identity map, transitivity

$$\_ \circ \_ : \mathtt{Id}_A(v,w) \to \mathtt{Id}_A(u,v) \to \mathtt{Id}_A(u,w)$$

corresponds to composition, and symmetry

$$\_^{-1} : \mathtt{Id}_A(u,v) \to \mathtt{Id}_A(v,u)$$

corresponds to taking the inverse. These operations satisfy the expected groupoid equations, but in general only *up to propositional equality*; e.g., $(p \circ q) \circ r$ (for appropriately typed p, q, r) is not definitional equal to $p \circ (q \circ r)$, but we can define a *higher* equality between equalities

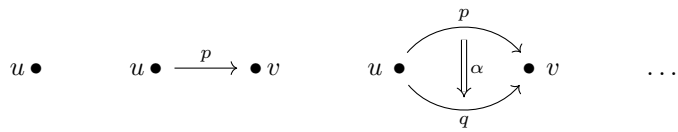
$$\alpha_{p,q,r} : \mathtt{Id}_{\mathtt{Id}_A(u,w')}((p \circ q) \circ r, p \circ (q \circ r)).$$

Already the groupoid interpretation suggests that a type in intensional type theory should be thought of more than merely a "set". Instead, a type should be thought of as a topological space—but up to *homotopy*. Around 2006 Awodey and Warren [9] and Garner [38] discovered connections between

[12] A category where each arrow is invertible.

homotopy theory and type theory in the context of Quillen model categories. Moreover, Streicher [76] and independently Voevodsky [54] built a model of type theory using Kan simplicial sets (the latter model also supporting type-theoretic universes, i.e., types of types).

One crucial point in this analogy between types and spaces is what under this view is the interpretation of the identity type: while $u : A$ should be thought of as a point in the space A, a term $p : \mathtt{Id}_A(u, v)$ should be thought of as a *path* between u and v, i.e., a continuous map $p\colon \mathbb{I} \to A$ (where $\mathbb{I} = [0, 1]$ is the interval) such that $p(0) = u$ and $p(1) = v$. Higher equalities $\alpha : \mathtt{Id}_{\mathtt{Id}_A(u,v)}(p, q)$ correspond to homotopies (or 2-dimensional paths) $\alpha\colon \mathbb{I} \to A^{\mathbb{I}} \cong \mathbb{I} \times \mathbb{I} \to A$ between the paths $p, q\colon \mathbb{I} \to A$ with $\alpha|_{\{0\}\times\mathbb{I}} = p$ and $\alpha|_{\{1\}\times\mathbb{I}} = q$; etc.



All the higher homotopies on a space A organize into a structure called the *fundamental ∞-groupoid* of A, the prime example of a so-called *weak ∞-groupoid*. These higher groupoids are closely connected to homotopy theory, as proposed by Grothendieck [44].

Another groundbreaking insight by Voevodsky was that the interpretation of type theory using Kan simplicial sets satisfies an additional axiom, the so-called *Univalence Axiom*. This axiom explains the identity type for type-theoretic universes. (A type-theoretic universe U is a type of types.) The Univalence Axiom states, loosely speaking, that isomorphic types (as elements of U) are equal, where "equal" refers to the identity type of U. This formalizes the everyday habit by mathematicians to identify isomorphic structures. Although this is clearly not valid in set theory: $0 \in \{0\}$ but $0 \notin \{1\}$ even though $\{0\} \cong \{1\}$. The Univalence Axiom can also be seen as an extensionality principle: it is a natural generalization of the concept in simple type theory formulated by Church [23] that two propositions are equal given that they are logically equivalent. The Univalence Axiom allows for a better encoding of mathematics in dependent type theory à la Martin-Löf, as is particularly shown in the book [79]. One aspect of this is that it entails the function extensionality axiom discussed above.

Another important contribution by Voevodsky is how notions from homotopy theory translate into the language of type theory. For example, the important hierarchy of *h-levels*: Call a type A *contractible*, or of h-level 0, if we have $(\Sigma a : A)(\Pi x : A)\ \mathtt{Id}_A(a, x)$. A type A is of h-level $n + 1$ if for all $x, y : A$ the type $\mathtt{Id}_A(x, y)$ is of h-level n. For a type to be of h-level 2 is then the same to have UIP for this type (those are called *h-sets*). Types with h-level 3 are called *h-groupoids*. The Univalence Axiom yields that this hierarchy is "well behaved", in particular it is closed under function types.

Similar to the situation with the function extensionality axiom, simply postulating the Univalence Axiom as a constant added to Martin-Löf type

theory destroys the good computational behavior of type theory, making it necessary to explain univalence computationally. One possible attempt to do so is to build a model of this axiom in type theory itself or at least in a constructive metatheory. Such a computational interpretation could then be obtained through semantics, for example, by evaluating a term of type N (the natural numbers) in the model.

The model of univalence using Kan simplicial sets by Voevodsky is, as it is, not suited to justify the axiom computationally since it is *not* constructive. This model is formulated using ZFC set theory and uses classical logic and the axiom of choice in an essential way. One problem with using simplicial sets has to do with the fact that the notion of degeneracy is *not decidable* in general and that simplicial maps have to commute with degeneracy maps. The theory of simplicial sets and Kan simplicial sets however crucially uses reasoning by cases on whether a simplex is degenerate or not. One example where this is needed is that for a Kan fibration, a path in the base induces an equivalence between the fibers over the endpoints; this was shown in [14], using a Kripke counter-model, to be not intuitionistically provable. Similar problems seem to appear when looking at the different proofs (e.g., [63, 40]) of the fact that B^A is a Kan simplicial set if B is so.

One possible remedy for this problem is to use Kan *semi*-simplicial sets instead as was done in [11]. This approach however is very involved and does not model various laws for substitutions. This part presents a different approach based on *cubical sets*. Cubical sets were used to give the first *combinatorial* definition of homotopy groups by Kan [53].

Our formulation of cubical sets gives a *formal* representation of cubes seen as continuous maps $u\colon \mathbb{I}^J \to X$ (with $\mathbb{I} = [0,1]$) where J is a finite set of names $x_1, \ldots, x_n$, instead of the more common $u\colon \mathbb{I}^n \to X$. We want to view such a cube as "value depending on the names" $x_1, \ldots, x_n$. We have face operations like, e.g., $u(x_i = 0)$, setting the x_i-coordinate to 0; for a fresh name y we can view u as depending on $x_1, \ldots, x_n, y$; this corresponds to the degeneracy operation; another primitive operation is to rename a variable. Thus the basic operations in cubical sets are certain substitutions of names. This formulation bears close resemblance to the theory of nominal sets [70, 69, 71]. There are various different variations of cubical sets used in the literature (see, e.g., [43]).

Formally cubical sets are given as presheaves on the (opposite of the) so-called cubical category, a category given by finite sets of names and certain substitutions between them. Following, e.g., [47], this yields a model of type theory where the contexts are interpreted as cubical sets. However to obtain the envisaged identity types we have to strengthen, similarly as in the Kan simplicial set model, the interpretation of types: we require types to have a *Kan structure*. This structure is a refinement of Kan's original extension condition ("Kan cubical sets") as defined in [53] which amounts to have fillers for all open boxes. We refine Kan's notion in two aspects: first, we require these fillers to be explicitly given as operations, and, second, that these operations satisfy certain uniformity conditions which ensure that the filling operations commute with the name substitutions in a suitable way. The second refinement

is crucially used to show that types are closed under dependent function spaces.

This part presents this model for type theory with identity types, Σ-types, Π-types, and a universe. The interpretation of the identity type, however, only satisfies the usual equation of the J-eliminator up to propositional equality and *not* as definitional equality as is usually required in type theory. The part is based on the publication [15] and adds more detailed proofs. A new contribution is the semantics of universes as Kan cubical sets. The treatment of the Univalence Axiom is not included here, but will be part of a future publication; sketches of the verification of univalence in this model are however given in [15, Section 8.4] and [28].

Moreover, this model (in its nominal set presentation) has been, implemented as the type-checker Cubical [25][13] together with C. Cohen, T. Coquand, and A. Mörtberg. This implementation builds on top of ordinary dependent type theory (without identity types) where certain primitive constants (giving rise to the properties we want for propositional equality) are available; whenever the type checker requires to check for conversion of two terms (i.e., definitional equality) we compare their semantics in the model. The implementation supports computing with the Univalence Axiom and in particular transporting along an equivalence.

Outline

In Chapter 1 we define semantics of Martin-Löf type theory relying on the notion of categories with families; we show how presheaves form an instance of this structure and thus give rise to a model of type theory. Chapter 2 introduces cubical sets as presheaves on the so-called cubical category along with examples. We also look into the relationship to nominal sets. Chapter 3 is the heart of this part: we introduce the notion of (uniform) Kan cubical sets and show that these induce a category with families extending the presheaf semantics. We give the interpretation of dependent sums, dependent products, and identity types. We also show how any cubical set can be "completed" to a Kan cubical set. In Chapter 4 we give the construction of a universe of Kan cubical set and show that it is itself a Kan cubical set, thus providing the semantics of a type-theoretic universe in our model.

[13] The version relevant for this part is on the `master` branch dated September 24, 2014.

Chapter 1

Semantics of Martin-Löf Type Theory

In this chapter we will introduce semantics of type theory based on the notion of categories with families. As an extended example we explain how presheaf categories induce such a model of type theory. This is the basis for the model described in the next chapters.

1.1 Categories with Families

There are various similar notions to organize models of dependent type theory. In what follows we chose categories with families (CwF) which were introduced by Dybjer [34] and further popularized by Hofmann [47]. Categories with families can be seen as an *algebraic* presentation of type theory. Even though the definition of a CwF is given using categorical language we want to stress the fact that it is an instance of a generalized algebraic theory [22]. To devise a CwF is to give: interpretations (as sets) for the sorts of *contexts*, *context morphisms*, *types*, and *terms*; operations including the context extension; and to check equations involving those operations.

We will not be concerned with the (non-trivial) task to interpret the syntax of Martin-Löf type theory into a CwF. We refer the reader to [47] for a sketch of this.

The category of families of sets, **Fam**, has as objects (A, B) where A is a set and $B = (B_a \mid a \in A)$ is a A-indexed family of sets B_a, $a \in A$. A morphism between $(A, B) \to (A', B')$ is given by a pair (f, g) where $f \colon A \to A'$ is a function and g is an A-indexed family of functions such that $g_a \colon B_a \to B'_{fa}$.

Definition 1.1.1. A *category with families* (CwF for short) is given by $(\mathcal{C}, \mathcal{F})$ where:

1. $\mathcal{C}$ is a category whose objects $\Gamma, \Delta, \ldots$ we call *contexts* and whose morphisms $\sigma, \tau, \ldots$ we call *substitutions* or *context morphisms*; we write $\Gamma \vdash$

to indicate that Γ is an object of $\mathcal{C}$, if $\mathcal{C}$ is clear from the context.

2. A terminal object 1 in $\mathcal{C}$ called the *empty context*.

3. $\mathcal{F}$ is a functor $\mathcal{F}\colon \mathcal{C}^{\mathrm{op}} \to \mathbf{Fam}$; for $\Gamma \vdash$ we write $\mathrm{Ty}_{\mathcal{F}}(\Gamma)$ for the indexing set of the family $\mathcal{F}(\Gamma)$ and its family as $(\mathrm{Ter}_{\mathcal{F}}(\Gamma; A) \mid A \in \mathrm{Ty}_{\mathcal{F}}(\Gamma))$; we also write $\Gamma \vdash A$ for $A \in \mathrm{Ty}_{\mathcal{F}}(\Gamma)$ and call A a *type in context* Γ (or over Γ), and $\Gamma \vdash a : A$ for $a \in \mathrm{Ter}_{\mathcal{F}}(\Gamma; A)$ and call a a term of type A in context Γ. For a substitution $\sigma\colon \Delta \to \Gamma$ the morphism $\mathcal{F}(\sigma)$ acts on types $\Gamma \vdash A$ as $A\sigma$, and on terms $\Gamma \vdash a : A$ as $a\sigma$. Note that $\Delta \vdash A\sigma$ and $\Delta \vdash a\sigma : A\sigma$, and the fact that $\mathcal{F}$ is a functor yields:

$$A\mathbf{1} = A \quad (A\sigma)\tau = A(\sigma\tau) \quad a\mathbf{1} = a \quad (a\sigma)\tau = a(\sigma\tau)$$

(Here and henceforth $\mathbf{1}$ denotes the suitable identity morphism.)

4. The operations of *context extension*: if $\Gamma \vdash$ and $\Gamma \vdash A$, there is a context $\Gamma.A$, a context morphism $\mathsf{p}\colon \Gamma.A \to \Gamma$, and a term $\Gamma.A \vdash \mathsf{q} : A\,\mathsf{p}$. This should satisfy the following (universal) property: for each $\Delta \vdash$ and substitution $\sigma\colon \Delta \to \Gamma$ and $\Delta \vdash a : A\sigma$, there is a substitution $(\sigma, a)\colon \Delta \to \Gamma.A$ such that:

$$\mathsf{p}(\sigma, a) = \sigma \qquad \mathsf{q}(\sigma, a) = a \qquad (\sigma, a)\tau = (\sigma\tau, a\tau) \qquad (\mathsf{p}, \mathsf{q}) = \mathbf{1}$$

where in the last equation $\mathbf{1}\colon \Gamma.A \to \Gamma.A$.

Note that above we use polymorphic notation to increase readability as in [22, 34]; e.g., without this convention we should have written $\mathsf{p}_{\Gamma,A}$ for the first projection $\mathsf{p}\colon \Gamma.A \to \Gamma$. We also leave parameters implicit, so, e.g., the equation $(A\sigma)\tau = A(\sigma\tau)$ tacitly assumes premises $\sigma\colon \Delta \to \Gamma$, $\tau\colon \Theta \to \Delta$, and $\Gamma \vdash A$.

Example 1.1.2. We can make the category of sets **Set** into the category of contexts of a CwF if we set the types over a set Γ to be the families of (small) sets A_γ indexed over $\gamma \in \Gamma$. A term is simply a dependent function $a_\gamma \in A_\gamma$ for each $\gamma \in \Gamma$, i.e., an element in the (set-theoretic) dependent function space $\Pi_{\gamma\in\Gamma} A_\gamma$. Context extensions are defined by the disjoint union

$$\Gamma.A = \{(\gamma, a) \mid \gamma \in \Gamma \text{ and } a \in A_\gamma\}$$

with $\mathsf{p}(\gamma, a) = a$ and $\mathsf{q}(\gamma, a) = a$.

As already indicated in the notation above we will usually suppress the reference to the CwF if clear from context. Moreover, we will usually present properties in rule form from now on. So, e.g., we will present the above

definition in rule form as:

$$\frac{}{1 \vdash} \qquad \frac{\Gamma \vdash \qquad \Gamma \vdash A}{\Gamma.A \vdash}$$

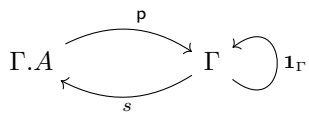
$$\frac{\Gamma \vdash A \qquad \sigma : \Delta \to \Gamma}{\Delta \vdash A\sigma} \qquad \frac{\Gamma \vdash a : A \qquad \sigma : \Delta \to \Gamma}{\Delta \vdash a\sigma : A\sigma}$$

$$\frac{\Gamma \vdash A}{\mathsf{p} : \Gamma.A \to \Gamma} \qquad \frac{\Gamma \vdash A}{\Gamma.A \vdash \mathsf{q} : A\mathsf{p}}$$

$$\frac{\sigma : \Delta \to \Gamma \qquad \Gamma \vdash A \qquad \Delta \vdash a : A\sigma}{(\sigma, a) \colon \Delta \to \Gamma.A}$$

If we have a type $\Gamma \vdash A$ and a dependent type or term over A, say $\Gamma.A \vdash B$, the operation to substitute a term of A, say $\Gamma \vdash a \colon A$, is represented in a CwF as follows. We have the identity context morphism $\mathbf{1} \colon \Gamma \to \Gamma$ and hence we can form $(\mathbf{1}, a) \colon \Gamma \to \Gamma.A$ which we usually denote by $[a] \colon \Gamma \to \Gamma.A$; this gives a type $\Gamma \vdash B[a]$.

Remark 1.1.3. Terms $\Gamma \vdash a : A$ are in a one-to-one correspondence with sections $s \colon \Gamma \to \Gamma.A$ of $\mathsf{p} : \Gamma.A \to \Gamma$, i.e., $\mathsf{p}s = \mathbf{1}$,



and, moreover, all sections are of the form $[a]$ for some $\Gamma \vdash a : A$.

As the definition of CwF is an instance of a generalized algebraic theory [34, Section 2.2], there is a notion of morphism of CwFs: we have to give operators for each of the different sorts preserving the required equations. We will however not make use of this notion and refer the reader to [22, Section 11] for the precise definition.

A mere CwF does not give much structure and only models type dependencies. We are interested in CwFs that have more structure.

A CwF *supports* Π*-types* if it is closed under the following rules:

$$\frac{\Gamma \vdash A \qquad \Gamma.A \vdash B}{\Gamma \vdash \Pi AB} \qquad \frac{\Gamma.A \vdash b : B}{\Gamma \vdash \lambda\, b : \Pi AB} \qquad \frac{\Gamma \vdash u : \Pi AB \qquad \Gamma \vdash a : A}{\Gamma \vdash \mathsf{app}(u, a) : B[a]}$$

Furthermore, we require the operations to satisfy β- and η-laws

$$\begin{aligned} \mathsf{app}(\lambda\, b, a) &= b[a] \\ u &= \lambda\ \mathsf{app}(u\mathsf{p}, \mathsf{q}) \end{aligned}$$

and laws for commutation with substitutions:

$$\begin{aligned}(\Pi AB)\sigma &= \Pi(A\sigma)(B(\sigma\mathsf{p},\mathsf{q}))\\ (\lambda\, b)\sigma &= \lambda\,(b(\sigma\mathsf{p},\mathsf{q}))\\ \mathtt{app}(u,a)\sigma &= \mathtt{app}(u\sigma,a\sigma)\end{aligned}$$

Note that these equations only make sense in the appropriate types. For, e.g., $(\lambda\, b)\sigma = \lambda\,(b(\sigma\mathsf{p},\mathsf{q}))$ to make sense, we need the equation $(\Pi AB)\sigma = \Pi(A\sigma)(B(\sigma\mathsf{p},\mathsf{q}))$.

Likewise, a CwF *supports* Σ*-types* if it is closed under the following rules:

$$\frac{\Gamma\vdash A \qquad \Gamma.A\vdash B}{\Gamma\vdash \Sigma AB} \qquad\qquad \frac{\Gamma\vdash a:A \qquad \Gamma\vdash b:B[a]}{\Gamma\vdash (a,b):\Sigma AB}$$

$$\frac{\Gamma\vdash u:\Sigma AB}{\Gamma\vdash \mathsf{p}u:A} \qquad\qquad \frac{\Gamma\vdash u:\Sigma AB}{\Gamma\vdash \mathsf{q}u:B[\mathsf{p}u]}$$

Note that we overload the notation for pairs and projections in Σ-types with the notation for context morphisms as they are required to satisfy similar equations:

$$\begin{aligned}\mathsf{p}(a,b) &= a\\ \mathsf{q}(a,b) &= b\\ u &= (\mathsf{p}u,\mathsf{q}u)\end{aligned}$$

and the laws for substitutions:

$$\begin{aligned}(\Sigma AB)\sigma &= \Sigma(A\sigma)(B(\sigma\mathsf{p},\mathsf{q}))\\ (a,b)\sigma &= (a\sigma,b\sigma)\\ (\mathsf{p}u)\sigma &= \mathsf{p}(u\sigma)\\ (\mathsf{q}u)\sigma &= \mathsf{q}(u\sigma)\end{aligned}$$

Note that the above definitions for a CwF to support a type former are rather direct from the corresponding syntactical formulation. A morphism of CwFs between CwFs supporting a certain structure (like Π-types) preserves this structure if the type and term formers are preserved.

Since it is slightly easier to use later on, we use Paulin-Mohring's formulation of the identity type [68]. A CwF *supports identity types* if it is closed under the following rules:

$$\frac{\Gamma\vdash A \qquad \Gamma\vdash a:A \qquad \Gamma\vdash b:A}{\Gamma\vdash \mathtt{Id}_A(a,b)} \qquad\qquad \frac{\Gamma\vdash a:A}{\Gamma\vdash \mathtt{refl}\,a:\mathtt{Id}_A(a,a)}$$

$$\frac{\begin{array}{c}\Gamma\vdash A \qquad \Gamma\vdash a:A \qquad \Gamma.A.\,\mathtt{Id}_{A\mathsf{p}}(a\mathsf{p},\mathsf{q})\vdash C\\ \Gamma\vdash v:C[a,\mathtt{refl}\,a] \qquad \Gamma\vdash b:A \qquad \Gamma\vdash p:\mathtt{Id}_A(a,b)\end{array}}{\Gamma\vdash \mathtt{J}(a,v,b,p):C[b,p]}$$

Where $[b,p]$ is the substitution $([b],p)\colon \Gamma \to \Gamma.A.\,\mathtt{Id}_{A\mathsf{p}}(a\mathsf{p},\mathsf{q})$. As before, we require that the operations commute with substitution:

$$\begin{aligned}(\mathtt{Id}_A(a,b))\sigma &= \mathtt{Id}_{A\sigma}(a\sigma,b\sigma)\\ (\mathtt{refl}\,a)\sigma &= \mathtt{refl}(a\sigma)\\ (\mathtt{J}(a,v,b,p))\sigma &= \mathtt{J}(a\sigma,v\sigma,b\sigma,p\sigma)\end{aligned}$$

where in the last equation $\mathtt{J}$ on the right hand side is w.r.t. C, and on the left hand side w.r.t. $C(\sigma\mathsf{p},\mathsf{q})$. (Note that $\mathtt{J}$ depends on C although suppressed in our syntax.) Additionally, we require

$$\mathtt{J}(a,v,a,\mathtt{refl}\,a) = v. \tag{1.1}$$

The identity type of the model we consider later in Chapter 3 will not satisfy equation (1.1). For this reason we say that a CwF supports *weak identity types* if all conditions of identity types except equation (1.1) are satisfied but where equation (1.1) holds only propositionally, i.e., only up to a witness of a respective identity type, that is, we require the rule

$$\frac{\Gamma\vdash A \qquad \Gamma\vdash a:A \qquad \Gamma.A.\,\mathtt{Id}_{A\mathsf{p}}(a\mathsf{p},\mathsf{q})\vdash C \qquad \Gamma\vdash v : C[a,\mathtt{refl}\,a]}{\Gamma\vdash \mathtt{JEq}(a,v) : \mathtt{Id}_{C[a,\mathtt{refl}\,a]}(v,\mathtt{J}(a,v,a,\mathtt{refl}\,a))}$$

such that $(\mathtt{JEq}(a,v))\sigma = \mathtt{JEq}(a\sigma,v\sigma)$.

There are usually two different ways to introduce a universe to dependent type theory: universes à la Tarski and universes à la Russell. A universe à la Tarski contains *codes* of the actual types and is equipped with a function decoding codes into types, where as a universe à la Russell has *types* as its elements. We will use a variant of a universe à la Tarski but in a formulation which is less general but simpler and is sufficient for the universes we consider (cf. Section 1.2.4).

Definition 1.1.4. A *universe* in a CwF $(\mathcal{C},\mathcal{F})$ is a CwF $\mathcal{U}=(\mathcal{C},\mathcal{F}_0)$ on the same category of contexts and substitutions such that $\mathrm{Ty}_{\mathcal{F}_0}(\Gamma)\subseteq \mathrm{Ty}_{\mathcal{F}}(\Gamma)$ and $\mathrm{Ter}_{\mathcal{F}_0}(\Gamma;A)=\mathrm{Ter}_{\mathcal{F}}(\Gamma;A)$ if $A\in\mathrm{Ty}_{\mathcal{F}_0}(\Gamma)$, and the context operations of $\mathcal{U}$ are inherited from $(\mathcal{C},\mathcal{F})$. We write $\Gamma\vdash A\ \mathsf{Type}_0$ for $A\in\mathrm{Ty}_{\mathcal{F}_0}(\Gamma)$ and call A a *$\mathcal{U}$-small type*; thus we require:

$$\frac{\Gamma\vdash A\ \mathsf{Type}_0}{\Gamma\vdash A}$$

Moreover, we require that there is a type $1\vdash U$ (writing also U for $U\sigma$ for the (unique) substitution $\sigma\colon\Gamma\to 1$) equipped with coding and decoding functions

$$\frac{}{1\vdash U} \qquad \frac{\Gamma\vdash A\ \mathsf{Type}_0}{\Gamma\vdash \ulcorner A\urcorner : U} \qquad \frac{\Gamma\vdash T:U}{\Gamma\vdash \mathrm{El}\,T\ \mathsf{Type}_0}$$

$$U\sigma = U \qquad \ulcorner A\urcorner\sigma = \ulcorner A\sigma\urcorner \qquad (\mathrm{El}\,T)\sigma = \mathrm{El}(T\sigma)$$

satisfying the equations

$$\mathrm{El}\ulcorner A\urcorner = A \quad \text{and} \quad \ulcorner \mathrm{El}\, T\urcorner = T. \tag{1.2}$$

If $(\mathcal{C}, \mathcal{F})$ supports Π-types, we say that the universe U supports Π-types if $(\mathcal{C}, \mathcal{F}_0)$ is closed under the induced Π-types, or in other words if U supports Π-types and the inclusion CwF-morphism $\mathcal{U} \hookrightarrow (\mathcal{C}, \mathcal{F})$ preserves Π. Similarly for other type formers.

Note with the equations (1.2) there is no need to require coding functions for the type formers which simplifies the treatment of universes significantly. E.g., for Π-types given $\Gamma \vdash a : U$ and $\Gamma.\,\mathrm{El}\, a \vdash b : U$ we can define $\Gamma \vdash \pi\ a\ b : U$ by

$$\pi\ a\ b = \ulcorner \Pi(\mathrm{El}\, a)(\mathrm{El}\, b)\urcorner$$

which satisfies $\mathrm{El}(\pi\ a\ b) = \Pi(\mathrm{El}\, a)(\mathrm{El}\, b)$.

1.2 Presheaf Models of Type Theory

We will now show how any presheaf category gives rise to a category with families where the contexts are presheaves. Let us first recall the notion of presheaf.

Definition 1.2.1. Let $\mathcal{C}$ be a category. A *presheaf* on $\mathcal{C}$ is a contravariant functor from $\mathcal{C}$ into **Set**. The category of presheaves on $\mathcal{C}$, denoted by $\mathrm{Psh}(\mathcal{C})$ or sometimes $\hat{\mathcal{C}}$, is the functor category $[\mathcal{C}^{\mathrm{op}}, \mathbf{Set}]$. In particular its morphisms are natural transformations.

Let us now fix a small category $\mathcal{C}$. In this section, we denote objects of $\mathcal{C}$ by I, J, K and morphisms by f, g, h. In what follows we describe how $\mathrm{Psh}(\mathcal{C})$ induces a CwF where the category of contexts is $\mathrm{Psh}(\mathcal{C})$.

So a context $\Gamma \vdash$ is a presheaf Γ on $\mathcal{C}$ is a functor $\Gamma\colon \mathcal{C}^{\mathrm{op}} \to \mathbf{Set}$, i.e., we are given a set $\Gamma(I)$ for each I in $\mathcal{C}$, and functions $\Gamma(I) \to \Gamma(J), \rho \mapsto \rho f$ (called *restriction*, and written as f acting on the right) for each $f\colon J \to I$, such that

$$\rho\mathbf{1} = \rho \quad \text{and} \quad (\rho f)g = \rho(fg)$$

for $g\colon K \to J$ and $f\colon J \to I$. In this section we will sometimes refer to the map $\rho \mapsto \rho f$ as Γ_f.

The empty context 1 is the terminal presheaf which is constant a singleton $\{\star\}$.

A context morphism σ between contexts Γ and Δ is then a natural transformation $\sigma\colon \Delta \to \Gamma$, i.e., for each I in $\mathcal{C}$ there is a map $\sigma_I\colon \Delta(I) \to \Gamma(I)$ such that for any $f\colon J \to I$ the square

$$\begin{array}{ccc} \Delta(I) & \xrightarrow{\sigma_I} & \Gamma(I) \\ {\scriptstyle \Delta_f}\downarrow & & \downarrow{\scriptstyle \Gamma_f} \\ \Delta(J) & \xrightarrow{\sigma_J} & \Gamma(J) \end{array}$$

commutes, i.e., $\Gamma_f \circ \sigma_I = \sigma_J \circ \Delta_f$. From now on we will suppress writing the subscripts to σ; this way, if we write Γ_f and Δ_f as f acting on the right again, the equation simply becomes

$$(\sigma\rho)f = \sigma(\rho f)$$

for ρ in $\Delta(I)$.

Next, we describe how to give a dependent type $\Gamma \vdash A$ in a context $\Gamma \vdash$. For each object $I \in \mathcal{C}$ and $\rho \in \Gamma(I)$ we require a set $A\rho$, and for each $f\colon J \to I$, we require a function $A\rho \to A(\rho f)$, written as $a \mapsto af$ satisfying $a\mathbf{1} = a$ and $afg = a(fg)$ if $g\colon K \to J$. Note that we tacitly suppressed the dependence on I in $A\rho$ in our notation to keep it lighter; similarly we omit I and ρ in af. Substitution $\Delta \vdash A\sigma$ with $\sigma\colon \Delta \to \Gamma$ is simply given $(A\sigma)\rho = A(\sigma\rho)$ for $\rho \in \Delta(I)$, together with the induced map

$$(A\sigma)\rho = A(\sigma\rho) \to A((\sigma\rho)f) = A(\sigma(\rho f)) = (A\sigma)(\rho f)$$

for $f\colon J \to I$. Clearly, this satisfies the required equations for substitutions.

Note that types in the empty context $1 \vdash A$ correspond exactly to contexts $A \vdash$. We will usually write $\vdash A$ instead of $1 \vdash A$.

The definition of a dependent type can also be rephrased more categorically: A is a presheaf on $\int_{\mathcal{C}} \Gamma$, where $\int_{\mathcal{C}} \Gamma$ is the *category of elements* of the presheaf Γ, defined as follows:

- objects are pairs (I, ρ) where $I \in \mathcal{C}$ and $\rho \in \Gamma(I)$;
- a morphism $(J, \rho') \to (I, \rho)$ is a morphism $f\colon J \to I$ in $\mathcal{C}$ such that $\rho' = \Gamma_f \rho$.

Note that substitution of $A\colon (\int_{\mathcal{C}} \Gamma)^{\mathrm{op}} \to \mathbf{Set}$ with $\sigma\colon \Delta \to \Gamma$ corresponds to precomposing with $\int_{\mathcal{C}} \sigma\colon \int_{\mathcal{C}} \Delta \to \int_{\mathcal{C}} \Gamma$ induced by σ. (In fact, this construction induces a functor $\int_{\mathcal{C}}\colon \mathrm{Psh}(\mathcal{C}) \to \mathbf{Cat}$.)

A term $\Gamma \vdash a : A$ of a dependent type $\Gamma \vdash A$ is given by a family of elements $a\rho \in A\rho$ for each I in $\mathcal{C}$ and $\rho \in \Gamma(I)$, such that $a\rho f = a(\rho f)$ for each $f\colon J \to I$. The substitution $\Delta \vdash a\sigma : A\sigma$ with $\sigma\colon \Delta \to \Gamma$ is given by the family $(a\sigma)\rho = a(\sigma\rho)$.

For $\Gamma \vdash A$, the context extension $\Gamma.A \vdash$ is defined by

$$\begin{aligned} (\Gamma.A)(I) &= \{(\rho, u) \mid \rho \in \Gamma(I) \text{ and } u \in A\rho\} && \text{for } I \in \mathcal{C} \\ (\rho, u)f &= (\rho f, uf) && \text{for } f\colon J \to I \end{aligned}$$

and the projections are defined by $\mathsf{p}\colon \Gamma.A \to \Gamma$, $\mathsf{p}(\rho, u) = \rho$, and $\Gamma.A \vdash \mathsf{q} : A\mathsf{p}$ by $\mathsf{q}(\rho, u) = u$. Now assume $\sigma\colon \Delta \to \Gamma$ and $\Delta \vdash a : A\sigma$; we define $(\sigma, a)\colon \Delta \to \Gamma.A$ by $(\sigma, a)\rho = (\sigma\rho, a\rho)$. One readily checks that this satisfies all the required equations; this concludes the definition of the CwF associated to a presheaf category.

For later use let us recall the Yoneda Lemma: the Yoneda embedding is the functor $\mathbf{y}\colon \mathcal{C} \to \hat{\mathcal{C}}$ is given by $\mathbf{y}I = \mathrm{Hom}_{\mathcal{C}}(-, I)$, i.e.,

$$\begin{aligned}&(\mathbf{y}I)(J) = \mathrm{Hom}_{\mathcal{C}}(J, I) \quad \text{for } I \text{ an object of } \mathcal{C}\\ &(\mathbf{y}I)_f\colon (\mathbf{y}I)(J) \to (\mathbf{y}I)(K), g \mapsto fg \quad \text{for } f\colon K \to J \text{ in } \mathcal{C}.\end{aligned}$$

The Yoneda functor is fully faithful and we have for a presheaf Γ,

$$\Gamma(I) \cong \mathrm{Hom}_{\mathrm{Psh}(\mathcal{C})}(\mathbf{y}I, \Gamma)$$

both natural in Γ and I. A presheaf Γ is representable if it is naturally isomorphic to a $\mathbf{y}I$ for some I.

1.2.1 Dependent Products

As a motivation for the definition of dependent products let us recall how to define exponents in presheaf categories. Let Γ and Δ be presheaves and suppose we already know how the exponent Δ^Γ is constructed. Then we get by the Yoneda Lemma and the fact that $-^\Gamma$ should be right adjoint to $- \times \Gamma$, that

$$\begin{aligned}(\Delta^\Gamma)(I) &\cong \mathrm{Hom}(\mathbf{y}I, \Delta^\Gamma)\\ &\cong \mathrm{Hom}(\mathbf{y}I \times \Gamma, \Delta)\end{aligned}$$

The latter can now be taken *as a definition*. So an element w of $(\Delta^\Gamma)(I)$ is a natural transformation $w\colon \mathbf{y}I \times \Gamma \to \Delta$, so it is given by functions

$$w_J\colon (\mathbf{y}I)(J) \times \Gamma(J) \to \Delta(J),$$

and the naturality condition becomes $(w_J(f, \rho))g = w_K(fg, \rho g)$ for $f\colon J \to I$, $\rho \in \Gamma(J)$, and $g\colon K \to J$.

We will now show that the CwF associated to a presheaf category supports Π-types. Given $\Gamma \vdash A$ and $\Gamma.A \vdash B$ we have to define $\Gamma \vdash \Pi AB$, that is, we have to define the set $(\Pi AB)\rho$ for $I \in \mathcal{C}$ and $\rho \in \Gamma(I)$. The elements w of $(\Pi AB)\rho$ are families $w = (w_f \mid J \in \mathcal{C}, f\colon J \to I)$ of (dependent) functions such that

$$w_f u \in B(\rho f, u) \quad \text{for } J \in \mathcal{C}, f\colon J \to I, \text{ and } u \in A(\rho f),$$

with the requirement that for $g\colon K \to J$

$$(w_f u)g = w_{fg}(ug).$$

For such a family $w \in (\Pi AB)\rho$, the restriction $wf \in (\Pi AB)(\rho f)$ for $f\colon J \to I$ is defined by taking

$$(wf)_g u = w_{fg} u \in B(\rho fg, u)$$

where $g\colon K \to J$ and $u \in A(\rho fg)$. (Note that this needs $\rho fg = \rho(fg)$.) This definition satisfies $w\mathbf{1} = w$ and $wfg = w(fg)$.

Now let $\Gamma.A \vdash b : B$; we have to define $\Gamma \vdash \lambda\, b : \Pi AB$, that is, give $(\lambda\, b)\rho \in (\Pi AB)\rho$. For $f\colon J \to I$ and $u \in A(\rho f)$ we set

$$((\lambda\, b)\rho)_f u = b(\rho f, u) \in B(\rho f, u).$$

This satisfies for $g\colon K \to J$

$$(((\lambda\, b)\rho)_f u)g = (b(\rho f, u))g = b(\rho(fg), ug) = ((\lambda\, b)\rho)_{fg}(ug)$$

and defines a term since

$$(((\lambda\, b)\rho)f)_g u = ((\lambda\, b)\rho)_{fg} u = b(\rho fg, u) = ((\lambda\, b)(\rho f))_g u.$$

To define the application let $\Gamma \vdash u : \Pi AB$ and $\Gamma \vdash v : A$, and we set $\mathtt{app}(u, v)\rho = (u\rho)_{\mathbf{1}}(v\rho) \in B(\rho, v\rho)$, thus $\mathtt{app}(u, v)\rho \in B[v]\rho$ as $(\rho, v\rho) = (\mathbf{1}, v)\rho = [v]\rho$. β-equality is readily checked

$$\mathtt{app}(\lambda\, b, v)\rho = ((\lambda\, b)\rho)_{\mathbf{1}}(v\rho) = b(\rho, v\rho) = b[v]\rho,$$

and similarly for η-equality and the other equations.

It is also possible to calculate the dependent product using the Yoneda Lemma similar to exponents (i.e., non-dependent products); this does however not add much to the explanation so we have refrained from adding it.

Remark 1.2.2. It is worthwhile to note that there is a simpler way to describe the sections of Π-types. Given a section $\Gamma \vdash w : \Pi AB$ it satisfies $(w\rho)_f = ((w\rho)f)_{\mathbf{1}} = (w(\rho f))_{\mathbf{1}}$ by definition. This entails that w is determined by the $(w\rho)_{\mathbf{1}}$'s and moreover we have

$$((w\rho)_{\mathbf{1}} a)f = (w\rho)_f(af) = (w(\rho f))_{\mathbf{1}}(af).$$

Conversely, assume that we have a family φ of functions $\varphi\rho$ such that $\varphi\rho\, a \in B(\rho, a)$ for $a \in A\rho$ satisfying

$$(\varphi\rho\, a)f = \varphi(\rho f)(af).$$

This defines a section $\Gamma \vdash v : \Pi AB$ by putting

$$(v\rho)_f a = \varphi(\rho f)a.$$

These assignments are inverse to each other. Using this representation, application can be simply written as

$$\mathtt{app}(\varphi, v)\rho = \varphi\rho(v\rho),$$

and abstraction as $(\lambda\, u)\rho\, a = u(\rho, a)$.

1.2.2 Dependent Sums

The interpretation Σ-types $\Gamma \vdash \Sigma AB$ for $\Gamma \vdash A$ and $\Gamma.A \vdash B$ is defined by

$$(\Sigma AB)\rho = \{(a, b) \mid a \in A\rho \text{ and } b \in B(\rho, a)\}$$

for $\rho \in \Gamma(I)$, $I \in \mathcal{C}$. The restrictions are defined componentwise $(a, b)f = (af, bf)$ for $f \colon J \to I$. The pairing operation $\Gamma \vdash (u, v) : \Sigma AB$ of $\Gamma \vdash u : A$ and $\Gamma.A \vdash v : B[u]$ is defined by the componentwise pairing: $(u, v)\rho = (u\rho, v\rho)$. Likewise for the projections p and q, $(\mathsf{p}w)\rho = a$ and $(\mathsf{q}w)\rho = b$ for $w\rho = (a, b)$. This validates the necessary equations.

1.2.3 Identity Types

There is also a "standard" interpretation for identity types in a presheaf model. However, this is *not* the interpretation we are interested in since it is proof irrelevant and satisfies uniqueness of identity proofs. Thus we will only briefly sketch the definition: for $\Gamma \vdash A$, $\Gamma \vdash a : A$, and $\Gamma \vdash b : A$, the identity type is given by $(\mathtt{Id}_A(a, b))\rho = \{* \mid a\rho = b\rho\}$. This allows an interpretation for the rules of identity types, e.g., $(\mathtt{refl}\, a)\rho = *$; this interpretation is such that the equation (1.1) on page 25 holds. In the rest of this part, we will not make further use of this identity type.

1.2.4 Universes

We will now show how to lift a universe à la Grothendieck in the underlying set theory to the presheaf model following [48, 74].

Assume a universe of small sets, say $\mathbf{Set}_0 \in \mathbf{Set}$, and call the elements small sets. From this we will now give a type-theoretic universe. We recall from Definition 1.1.4 that to give a type-theoretic universe we have to first single out the small types: the judgment $\Gamma \vdash A\ \mathsf{Type}_0$ is defined to mean that for each $\rho \in \Gamma(I)$, the set $A\rho$ is small, i.e., $A\rho \in \mathbf{Set}_0$. In this case we call A a small type (in context Γ). We clearly have that any small type is a type, i.e.:

$$\frac{\Gamma \vdash A\ \mathsf{Type}_0}{\Gamma \vdash A}$$

Since our underlying universe of small sets $\mathbf{Set}_0$ is closed under set-theoretic operations we get that the small types $\Gamma \vdash A\ \mathsf{Type}_0$ form itself a CwF of small types supporting the discussed type forming operations like Π, Σ, and are closed under substitution.

Next, we have to give the type of codes for small types in the empty context $\vdash U$. This is the same as giving a context $U \vdash$. The definition has to be such that for each small type $\Gamma \vdash A\ \mathsf{Type}_0$ there is a code $\Gamma \vdash \ulcorner A \urcorner : U$, and for each section of U, say $\Gamma \vdash T : U$, there is a small type $\Gamma \vdash \mathrm{El}\, T\ \mathsf{Type}_0$ satisfying:

$$\mathrm{El} \ulcorner A \urcorner = A \qquad\qquad \ulcorner \mathrm{El}\, T \urcorner = T$$

We now define U as a context. For $I \in \mathcal{C}$, $U(I)$ consists of *small* types

$$\mathbf{y}I \vdash A\ \mathsf{Type}_0$$

where $\mathbf{y}$ denotes the Yoneda embedding. The restrictions are as well given by the Yoneda embedding: if $f\colon J \to I$ and $A \in U(I)$, that is, $\mathbf{y}I \vdash A$ is small, then the restriction $Af \in U(J)$ is defined to be the small type $\mathbf{y}J \vdash A\mathbf{y}f$, where $\mathbf{y}f : \mathbf{y}J \to \mathbf{y}I$.

Let us unfold the above definition: for $I \in \mathcal{C}$, $U(I)$ consists of small types $\mathbf{y}I \vdash A\ \mathsf{Type}_0$, that is, for each f with $\operatorname{cod} f = I$ a small set A_f (we use a subscript to not confuse the set A_f with the restriction of A along f written as Af), and A comes together with maps $A_f \to A_{fg}$, $a \mapsto ag$ for $g\colon K \to J$, $f\colon J \to I$ such that $a\mathbf{1} = a$ and $agh = a(gh)$ for $h\colon L \to K$. The restriction $U(I) \to U(J)$ along $f\colon J \to I$ is $(Af)_g = A_{\mathbf{y}fg} = A_{fg}$ for $g\colon K \to J$.

In other words, the elements of $U(I)$ can also be described as $\mathbf{Set}_0$-valued presheaves on $\mathcal{C}/I$ (the slice category over I), i.e., $[(\mathcal{C}/I)^{\mathrm{op}}, \mathbf{Set}_0]$, with the restriction induced by $\mathcal{C}/f : \mathcal{C}/J \to \mathcal{C}/I$. This is reflected by the equivalence of categories:

$$\mathcal{C}/I \approx \int_{\mathcal{C}} \mathbf{y}I$$

Next, we define a small type $\Gamma \vdash \mathrm{El}\, T$ given $\Gamma \vdash T : U$. For $\rho \in \Gamma(I)$ we have $T\rho \in U(I)$ and set $(\mathrm{El}\, T)\rho = (T\rho)_{\mathbf{1}_I}$ which is a small set. The required map $(\mathrm{El}\, T)\rho \to (\mathrm{El}\, T)\rho f$ for $f\colon J \to I$ is defined to be the given map $(T\rho)_{\mathbf{1}} \to (T\rho)_f$, which makes sense since $(\mathrm{El}\, T)\rho f = (T\rho f)_{\mathbf{1}} = ((T\rho)f)_{\mathbf{1}_J} = (T\rho)_f$.

If $\sigma\colon \Delta \to \Gamma$ and $\Gamma \vdash T : U$, then $(\mathrm{El}\, T)\sigma = \mathrm{El}(T\sigma)$ since for $\rho \in \Delta(I)$

$$((\mathrm{El}\, T)\sigma)\rho = (\mathrm{El}\, T)(\sigma\rho) = (T(\sigma\rho))_{\mathbf{1}} = ((T\sigma)\rho)_{\mathbf{1}} = (\mathrm{El}(T\sigma))\rho.$$

Last, we define the code $\Gamma \vdash \ulcorner A \urcorner : U$ of a small type $\Gamma \vdash A\ \mathsf{Type}_0$. For $\rho \in \Gamma(I)$ we define $\ulcorner A \urcorner \rho \in U(I)$ as the small type $\mathbf{y}I \vdash \ulcorner A \urcorner \rho$ given by $(\ulcorner A \urcorner \rho)_f = A(\rho f)$ for $f \in (\mathbf{y}I)(J)$, i.e., $f\colon J \to I$; and the induced restriction maps

$$(\ulcorner A \urcorner \rho)_f = A(\rho f) \to A(\rho f)g = A(\rho(fg)) = (\ulcorner A \urcorner \rho)_{fg}.$$

The verification of $\ulcorner A \urcorner \sigma = \ulcorner A\sigma \urcorner$ for a substitution $\sigma\colon \Delta \to \Gamma$ is straightforward.

It remains to check the equation (1.2) of Definition 1.1.4: we have that

$$(\mathrm{El} \ulcorner A \urcorner)\rho = (\ulcorner A \urcorner \rho)_{\mathbf{1}} = A(\rho\mathbf{1}) = A\rho,$$

and likewise

$$(\ulcorner \mathrm{El}\, T \urcorner \rho)_f = (\mathrm{El}\, T)(\rho f) = (T(\rho f))_{\mathbf{1}} = (T\rho)_f.$$

Analogously, one can lift a hierarchy of Grothendieck universes $\mathbf{Set}_0 \in \mathbf{Set}_1 \in \cdots \in \mathbf{Set}_n \in \cdots \in \mathbf{Set}$ to a type-theoretic hierarchy of universes $U_0, U_1, \ldots, U_n, \ldots$ with corresponding coding and decoding machinery.

Chapter 2

Cubical Sets

This chapter introduces cubical sets and discusses their relationship to nominal sets. Cubical sets are presheaves on the cubical category $\square$ introduced in Section 2.1.

2.1 The Cubical Category

We fix a countably infinite and discrete set of atomic names $\mathbb{A}$ which will serve as explicit names for dimensions. We will denote elements of $\mathbb{A}$ by $x, y, z, \ldots$ and call them *names*. Later we will also assume that given a *finite* set of names $I \subseteq \mathbb{A}$ there is a fresh name x_I. The choice of the set $\mathbb{A}$ is irrelevant—what counts is that we can decide the equality of names. We will also assume that there are two elements 0 and 1 called *directions* which are *not* names; we will usually use c, d to denote directions; we write $\bar{c}$ for flipping the direction, i.e., $\bar{c} = 1 - c$. We set $\mathbf{2} = \{0, 1\}$.

Definition 2.1.1. The *cubical category* $\square$ has as objects finite subsets of the fixed set of names $\mathbb{A}$, usually denoted by $I, J, K, \ldots$. A morphisms $f \colon I \to J$ is given by a set-theoretic function $f \colon I \to J \cup \mathbf{2}$ such that for $x, y \in I$ with $f\,x, f\,y \notin \mathbf{2}$ we have

$$f\,x = f\,y \text{ implies } x = y,$$

i.e., f is injective when restricted to its *defined elements*

$$\mathrm{def}(f) = \{x \in I \mid f\,x \notin \mathbf{2}\}.$$

The composition of two morphism $f \colon I \to J$ and $g \colon J \to K$ is given by

$$(g \circ f)\,x = \begin{cases} g(fx) & \text{if } x \in \mathrm{def}(f), \\ fx & \text{otherwise.} \end{cases}$$

For the composition of two morphisms f and g as above we also write $fg = g \circ f$, i.e., we use *diagram order*. For finite sets of names we will write

commas instead of unions and often omit curly braces; e.g., we write I, x, y for $I \cup \{x, y\}$, and $I - x, y$ for the set-difference of I with $\{x, y\}$. From now on, if not stated otherwise, we assume that f, g, h range over morphisms in $\square$.

The *face maps* are morphisms $(x = 0), (x = 1) \colon I \to I - x$ for $x \in I$ sending x to 0 and 1, respectively; so in particular $\mathrm{def}\,(x = c) = I - x$. For $I \subseteq J$, we call the inclusion map $I \to J$ a *degeneracy map*; in particular, if $x \notin I$, we denote the inclusion $I \subseteq I, x$ by $s_x \colon I \to I, x$. For $x, y \notin I$ the *renaming* $(x = y) \colon I, x \to I, y$ sends x to y and leaves the rest untouched. (Note that it is crucial that both x and y are not in I, otherwise the injectivity condition on defined elements is violated.) For $x, y \in I$ the *swapping* of x and y, $(x\ y) \colon I \to I$, is defined by

$$(x\ y)z = \begin{cases} y & \text{if } z = x, \\ x & \text{if } z = y, \\ z & \text{otherwise.} \end{cases}$$

Note that with the standard programming trick to swap two variables using assignments and a third variable, we can write a swap $(x\ y) \colon I, x, y \to I, x, y$ (where $x, y \notin I$) as a composition of renamings: with a fresh z we have

$$\begin{array}{ccc} I, x, y & \xrightarrow{(x\ y)} & I, x, y \\ {\scriptstyle (x=z)}\downarrow & & \uparrow{\scriptstyle (z=y)} \\ I, y, z & \xrightarrow{(y=x)} & I, x, z \end{array}$$

For $f \colon I \to J$ set $f - x \colon I - x \to J - fx$ to be f restricted to $I - x$ with adapted codomain (where $J - fx = J$ if fx is 0 or 1). Similarly we can extend $f \colon I \to J$ for $x \notin I$ and a a name, 0, or 1, to $(f, x = a) \colon I, x \to J, a$ (by convention, $J, 0 = J, 1 = J$).

We call $f \colon I \to J$ *strict* if $\mathrm{def}(f) = I$.

The following lemma (whose proof is trivial) gives a factorization of any morphism as composition of faces, swapping (or renaming), and a degeneracy map.

Lemma 2.1.2. *Any $f \colon I \to J$ can be uniquely as $f = f_{01}g$ where $f_{01} \colon I \to \mathrm{def}(f)$ is a composition of face maps and $g \colon \mathrm{def}(f) \to J$ is a strict map. Moreover, if $I_0 = \vec{x} = f^{-1}(0)$ and $I_1 = \vec{y} = f^{-1}(1)$, then $f_{01} = (\vec{x} = 0)(\vec{y} = 1)$, and we have a commuting square*

$$\begin{array}{ccc} I & \xrightarrow{\ f\ } & J \\ {\scriptstyle f_{01}}\downarrow & \nearrow{\scriptstyle g} & \uparrow \\ \mathrm{def}(f) & \xrightarrow{\ f'\ } & \mathrm{im}(f) \end{array}$$

where f' is a bijection (and thus can be written a composition of transpositions, i.e., swappings, and thus also as a composition of renamings).

2.2 Cubical Sets

Similar to simplicial sets, cubical sets are defined as presheaves.

Definition 2.2.1. A *cubical set* X is a functor $X\colon \square \to \mathbf{Set}$, i.e., X is a presheaf over $\square^{\mathrm{op}}$.

That is, a cubical set X is given by sets $X(I)$ for each finite set of names I, and for each morphism in $f\colon I \to J$ in $\square$, a function $X(I) \to X(J)$, $u \mapsto uf$ such that

$$u\mathbf{1} = u \quad \text{and} \quad ufg = u(fg)$$

for $f\colon I \to J$ and $g\colon J \to K$ in $\square$. Note that the latter equation is the reason for the use of the diagram order for composition in $\square$; this enables us to write the action of a morphism on the right and still have the arrows from $\square$ and not its opposite category.

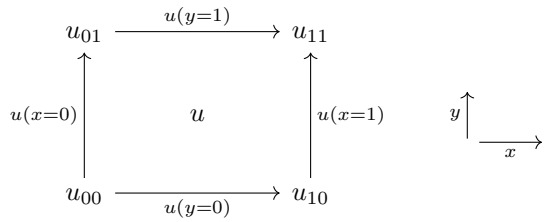
As with any presheaf category, the morphisms between cubical sets are given by natural transformations. This makes cubical sets into a category, denoted by **cSet**.

For $u \in X(I)$ and $x \in I$ we can form the faces $u(x = 0) \in X(I - x)$ and $u(x = 1) \in X(I - x)$ of u. In this way, we can think of u as a "line" connecting $u(x = 0)$ and $u(x = 1)$ along x, written as

$$u(x = 0) \xrightarrow[x]{u} u(x = 1) \tag{2.1}$$

We sometimes omit the subscript x in (2.1) if irrelevant or clear from the context.

In this way one can think of an element u in $X(I)$ as a hypercube of dimension $|I|$, and we call the elements of $X(I)$ also *I-cubes*. For example, if u in $X(x, y)$, $x \neq y$, first note that $u(x = c)(y = d) = u(y = d)(x = c)$ and thus with $u_{cd} = u(x = c)(y = d)$ we get a cube (indicating the naming of the dimensions on the right)



This cube should be thought as a "solid" cube, filled by u. Note that there is no "diagonal" in this cube, i.e., a face of u which connects u_{00} with u_{11}.

For an I-cube u and an $x \notin I$, we can consider the degenerate $us_x \in X(I, x)$ of u, connecting $us_x(x = 0) = u$ and $us_x(x = 1) = u$, i.e., u with itself along x:

$$u \xrightarrow[x]{us_x} u$$

Intuitively, one can think of us_x as the line which is constantly u. If $w = us_x$ for some u and x, we write that $w \# x$ borrowing notation from nominal sets (cf. Section 2.3) and say that w is *degenerate* (along x). Note that it is in general undecidable whether an element is degenerate.

Summing up, if we have an I-cube u we want to think of it as depending on the names in I; there are the following basic operations on u: renaming a name in I with a fresh name, setting one of the variables in I to 0 or 1, and adding a variable dependency (degeneracy maps).

From the fact that the category of cubical sets is a presheaf category we know that it has the structure of a topos and thus has a rich structure. In particular, we have seen in detail in Section 1.2 how **cSet** gives rise to a category with families. Let us introduce some examples of cubical sets.

Example 2.2.2. For every set A the *discrete cubical set* $\Delta(A)$ given by the constant presheaf $\Delta(A)(I) = A$ and $\Delta(A)(f) = \mathbf{1}_A$.

Example 2.2.3. A particularly natural example, suggested by Peter Aczel, is given by polynomial rings. Let R be a commutative ring with 1. For $I = x_1, \dots, x_n$ let $R[I] = R[x_1, \dots, x_n]$ denote the polynomial ring with indeterminates $x_1, \dots, x_n$ and coefficients in R. This assignment $I \mapsto R[I]$ defines a cubical set which we denote by $R[\cdot]$; if $f\colon I \to J$ and $p(x_1, \dots, x_n) \in R[I]$, pf is given by the polynomial $p(fx_1, \dots, fx_n)$. So, for example, $(1 + x^2y + z)(y = 0) = 1 + z$. Note that we assume $R[I] \subseteq R[I, x]$ and degeneracy is an inclusion: $ps_x = p$.

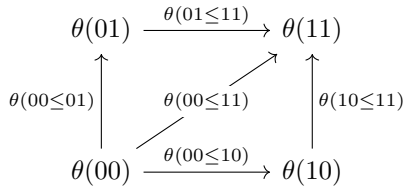
Example 2.2.4. The *interval* $\mathbb{I}$ is defined by $\mathbb{I}(I) = I \cup \mathbf{2}$ and $\mathbb{I}(f)\colon \mathbb{I}(I) \to \mathbb{I}(J)$ for $f\colon I \to J$ is defined by extending (the underlying set-theoretic map of) f with $\mathbb{I}(f)(0) = 0$ and $\mathbb{I}(f)(1) = 1$. Note that this is a representable cubical set: fix any name x, then $\mathbf{y}\{x\} \cong \mathbb{I}$. For example, $\mathbf{y}\{x\}\emptyset = \mathbf{cSet}(\{x\}, \emptyset)$ consists of the moprphisms $(x = 0), (x = 1)\colon x \to \emptyset$; similarly, $\mathbf{y}\{x\}\{y_1, \cdots, y_n\}$ consists of the morphisms $(x = a)$ for $a \in \mathbb{I}(y_1, \cdots, y_n)$.

More generally, we set $\square_I = \mathbf{y}I$ for a finite set of names I and call it the *standard I-cube*. For any I and J with n-elements, we clearly have $\square_I \cong \square_J$, and thus it makes sense to speak of $\square_I$ of a standard n-cube. The Yoneda Lemma gives that for a cubical set X, morphisms $\square_I \to X$ correspond to the elements of $X(I)$. Note that $\square_\emptyset$ is terminal in **cSet**, and thus $1 \to X$ corresponds to $X(\emptyset)$, the points of X.

Note that the product $\mathbb{I} \times \mathbb{I}$ is not isomorphic to $\square_{x,y}$. The latter does not contain the diagonal while the former does (cf. also the next example).

Example 2.2.5 (Nerve)**.** To any small category $\mathcal{C}$ we associate its *(cubical) nerve* $\mathrm{N}\mathcal{C}$ whose n-cubes are given by n-dimensional cubical commutative diagrams, defined as follows. For a finite set of names I we consider $\{0, 1\}^I = \mathbf{2}^I$ as a poset, and hence category. The *set* $(\mathrm{N}\mathcal{C})(I)$ is defined to be the functors $\mathbf{2}^I \to \mathcal{C}$. Any $f\colon I \to J$ determines a monotone $\mathbf{2}^f\colon \mathbf{2}^J \to \mathbf{2}^I$, sending $\alpha \in \mathbf{2}^J$ to $(\mathbf{2}^f\alpha)(i) = \alpha(fi)$ for $i \in I$ and f defined on i, and $(\mathbf{2}^f\alpha)(i) = fi$ otherwise; for $\theta \in (\mathrm{N}\mathcal{C})(I)$, i.e., $\theta\colon \mathbf{2}^I \to \mathcal{C}$ a functor, we set the restriction θf to be $\theta\mathbf{2}^f\colon \mathbf{2}^J \to \mathbf{2}^I \to \mathcal{C}$.

Elements of $(\mathrm{N}\mathcal{C})(\emptyset)$ are functors $1 \to \mathcal{C}$, i.e., correspond to objects in $\mathcal{C}$; elements of $(\mathrm{N}\mathcal{C})(x)$ are functors $\alpha\colon \{0,1\} \to \mathcal{C}$, i.e., correspond to morphisms in $\mathcal{C}$ (given by $\alpha(0 \leq 1)$, $0 \leq 1$ denoting the (unique) arrow $0 \to 1$ etc.); likewise, elements of $(\mathrm{N}\mathcal{C})(x,y)$ are functors $\theta\colon \mathbf{2}^{x,y} \to \mathcal{C}$, i.e., correspond to commuting squares in $\mathcal{C}$ (writing 01 for $(x \mapsto 0, y \mapsto 1) \in \mathbf{2}^{x,y}$ etc.)



and so on for higher cubes. Note that $\mathrm{N}(\mathbf{2}^I)$ is in general not isomorphic to $\mathbf{y}I$ (for $I = x, y$, the cube in the nerve contains a diagonal which is not there in the standard cube).

Remark 2.2.6. Often, cubical sets are described as presheaves on a category dual to our cubical category; morphisms are given by certain $\mathbf{2}^J \to \mathbf{2}^I$ (namely those which come from $\mathbf{2}^f$ with $f\colon I \to J$ in the cubical category). This is used (for a variation of the cubical sets considered here) in [43, Section 4].

The following two definitions are crucial for the rest of this part.

Definition 2.2.7 (Non-dependent Path Space)**.** Let X be a cubical set. The (non-dependent) *path space* $[\mathbb{I}]X$ is defined by $([\mathbb{I}]X)(I) = X(I, x_I)$ (recall that x_I is a chosen fresh name for I). We can extend $f\colon I \to J$ to $(f, x_I = x_J)\colon I, x_I \to J, x_J$, and thus define the restriction along f of $\omega \in ([\mathbb{I}]X)(I)$ by $\omega f = \omega(f, x_I = x_J)$. (Note that on the left hand side the restriction is in $[\mathbb{I}]X$ whereas on the right hand side it is in X.) This defines a cubical set. Its points are $([\mathbb{I}]X)(\emptyset) = X(x_\emptyset)$, that is, the lines of X; its lines $([\mathbb{I}]X)(x) = X(x, y)$, y fresh, are the squares of X; and so on.

There is also an alternative definition of $[\mathbb{I}]X$: the elements $([\mathbb{I}]X)(I)$ are (equivalence classes) of the form $\langle x\rangle p$ where $x \notin I$ and $p \in X(I, x)$; two such elements $\langle x\rangle p$ and $\langle y\rangle q$ get identified if $p(x = y) = q \in X(I, y)$. For $f\colon I \to J$ we define $(\langle x\rangle p)f = \langle z\rangle(p(f, x = z))$ where z is some J-fresh name. The operation $\langle x\rangle -$ should be thought of as an abstraction- or binding operation. Correspondingly, there is also an application of $\langle x\rangle p \in ([\mathbb{I}]X)(I)$ to $a \in \{0,1\}$ or a a fresh name (i.e., $a \notin I$) given by

$$(\langle x\rangle p) \mathbin{@} a = p(x = a) \in X(I, a),$$

where $I, 0 = I, 1 = I$ by convention. This structure can be seen as the "affine" exponential of X by $\mathbb{I}$ (this can be made precise in the presentation of cubical sets as nominal sets, cf. 2.4).

Our first definition corresponds to choosing a canonical representative x_I for the bound name. We will mostly use this definition from now on but deviate whenever appropriate. But note that we also have the operation $\omega \mathbin{@} a =$

$\omega(x_I = a) \in X(I, a)$ for $a \in \{0, 1\}$ or a fresh, with $\omega \in ([\mathbb{I}]X)(I)$, and the operation $\langle x \rangle p = p(x_I = x)$ for $p \in X(I, x)$.

Definition 2.2.8 (Non-dependent Identity Type)**.** For a cubical set X and two global sections $u, v \in X(\emptyset)$ we define the (non-dependent) *identity type* $\mathtt{Id}_X(u, v)$ to be the subobject $\mathtt{Id}_X(u, v) \subseteq [\mathbb{I}]X$ such that $\omega \in ([\mathbb{I}]X)(I)$ is an element of $(\mathtt{Id}_X(u, v))(I)$ if $\omega \mathbin{@} 0 = us_I$ and $\omega \mathbin{@} 1 = vs_I$, where $s_I \colon \emptyset \to I$ denotes the inclusion map; that is, ω is a line along x_I connecting u to v (more precisely, their degenerates).

More generally, we can also define $X \times X \vdash \mathtt{Id}_X$; for w_0 and w_1 in $X(I)$, $\mathtt{Id}_X(w_0, w_1)$ are those $\omega \in [\mathbb{I}]X$ such that $\omega \mathbin{@} 0 = w_0$ and $\omega \mathbin{@} 1 = w_1$. (The above cubical set $\mathtt{Id}_X(u, v)$ is given by substituting $X \times X \vdash \mathtt{Id}_X$ along $\langle u, v \rangle \colon 1 \to X \times X$.)

As an example, let us come back to the interval $\mathbb{I}$. We have two points $\vdash \mathtt{0} : \mathbb{I}$ and $\vdash \mathtt{1} : \mathbb{I}$ given by $\mathtt{0}_J = 0 \in \mathbb{I}(J)$ and likewise $\mathtt{1}_J = 1$. Those two points are equal via $\vdash \mathtt{seg} : \mathtt{Id}_{\mathbb{I}}(0, 1)$ given by $\mathtt{seg}_J = \langle x \rangle x$ since $x \in \mathbb{I}(J, x)$ for x J-fresh.

Remark 2.2.9. It is important to note that this definition (or its dependent version we will see later) does *not* justify the axioms for an identity type in the CwF induced by **cSet**. Let us illustrate one of the requirements; assume we have a cubical set X and a dependent type over it, $X \vdash C$, and two global sections of X, $\vdash u, v : X$ (which correspond to two elements $u, v \in X(\emptyset)$). Moreover, we have an $\vdash \omega : \mathtt{Id}_X(u, v)$ and $\vdash p : C[u]$. The rules for the identity type require in particular an inhabitant of $\vdash C[v]$, i.e., we must be able to transport the element p along ω to an element in $C[v]$; but for arbitrary C there is no hope that we can achieve this as $C[v]$ might be empty even though $C[u]$ is not! Consider, e.g., the type $\mathbb{I} \vdash C$ defined for $\rho \in \mathbb{I}(I) = I \cup \mathbf{2}$ as $C\rho = \emptyset$ if $\rho \neq 0$, and $C\rho = \mathbb{N}$ if $\rho = 0$. The restriction maps $C\rho \to C\rho f$ is given by the unique map $\emptyset \to C\rho f$ if $\rho \neq 0$, and given by the identity on $\mathbb{N}$ if $\rho = 0$ (and thus also $\rho f = 0$). Now $\mathtt{seg} = \langle x \rangle x$ gives a term of type $\mathtt{Id}_{\mathbb{I}}(0, 1)$, but there is no map from $C[0] = \mathbb{N}$ to $C[1] = \emptyset$. Hence we have to restrict the types in such a way that this property follows—this is the content of Chapter 3.

Remark 2.2.10. It is possible to justify the introduction rule for this identity types though: if $\vdash u : X$ given by the family $u_I \in X(I)$, define $\vdash \mathtt{refl}\, u : \mathtt{Id}_X(u, u)$ by $(\mathtt{refl}\, u)_I = u_I s_{x_I} \in X(I, x_I)$; note that $(\mathtt{refl}\, u)_I \in (\mathtt{Id}_X(u, u))(I)$ as

$$(\mathtt{refl}\, u)_I \mathbin{@} b = u_I.$$

Moreover, this defines a term as for $f \colon I \to J$,

$$(\mathtt{refl}\, u)_I f = (us_{x_I})f = us_{x_I}(f, x_I = x_J) = (uf)s_{x_J}.$$

2.3 Cubical Sets via Nominal Sets

In this section we give equivalent categories to the category of cubical sets which are given by nominal sets with extra structure: so called 01-substitutions introduced in [69]. As studied in [71], the latter can also be presented as finitely supported M-sets for a suitable monoid M. The theory of nominal sets provides a mathematical theory of *names* based on symmetry. For more background on nominal sets we refer the reader to [70]. Here we follow [71].

We want to think of I-cubes u in a cubical set X, for say $I = x_1, \dots, x_n$, as entities depending on the names $x_1, \dots, x_n$, and can emphasize this by writing $u = u(x_1, \dots, x_n)$ (similarly to indicate possible dependence of variables in a formula of predicate logic; see also Example 2.2.3). Now applying a morphism $f\colon I \to J$ should be thought of as applying a *substitution* of those names; the basic operations are renaming a variable into 0 or 1, renaming a variable into a fresh variable, and adding a (vacuous) variable dependency. E.g., if $n = 3$ and $f = (x_1 = y, x_2 = 0, x_3 = x_3)\colon I \to x_3, y, z$, we think of $uf \in X(x_3, y, z)$ as $u(y, 0, x_3)$ (with this notation, the application of a degeneracy map is not explicit).

Nominal sets capture this idea as well: each element in a nominal set depends on a *finite* set of names, and we can swap names (governed by suitable equations). The additional structure of 01-substitutions on a nominal set allows to set names to 0 or 1.

Definition 2.3.1. A *finite substitution* is a map $\pi\colon \mathbb{A} \to \mathbb{A} \cup \mathbf{2}$ such that $\{x \mid \pi x \neq x\}$ is finite. We denote the monoid of all finite substitutions by $\mathbf{Sb}$: its monoid operation is given by $\pi\pi'\colon \mathbb{A} \to \mathbb{A} \cup \mathbf{2}$ defined as $(\pi\pi')(x) = \pi'(\pi x)$ (note the order) if $\pi x \notin \mathbf{2}$ and $(\pi\pi')(x) = \pi x$ for $\pi x \in \mathbf{2}$; the unit 1 is given by the inclusion $\mathbb{A} \hookrightarrow \mathbb{A} \cup \mathbf{2}$. The element in $\mathbf{Sb}$ transposing x with y for $x, y \in \mathbb{A}$ is denoted by $(x\ y)$; the element $(x = b) \in \mathbf{Sb}$ for $x \in \mathbb{A}$ and $b \in \mathbf{2}$ sends x to b and is the identity otherwise.

Let us assume that $\mathbf{M}$ is a submonoid of $\mathbf{Sb}$.

Definition 2.3.2. The category of $\mathbf{M}$-*sets* is simply the presheaf category $[\mathbf{M}, \mathbf{Set}]$ where $\mathbf{M}$ is considered as a category (with one element). So such an $\mathbf{M}$-set Γ is given by a set Γ and an action $\Gamma \times \mathbf{M} \to \Gamma$ satisfying

$$\rho\, 1 = \rho \qquad (\rho\, \pi)\, \pi' = \rho\, (\pi\pi')$$

for $\rho \in \Gamma$ and $\pi, \pi' \in \mathbf{M}$.

A finite subset $I \subseteq \mathbb{A}$ *supports* an element $\rho \in \Gamma$ if for all $\pi, \pi' \in \mathbf{M}$,

$$\forall x \in I(\pi x = \pi' x) \to \rho\, \pi = \rho\, \pi'. \tag{2.2}$$

The *category of finitely supported* $\mathbf{M}$-*sets* $[\mathbf{M}, \mathbf{Set}]_{\mathsf{fs}}$ is the full subcategory of $\mathbf{M}$-sets for whose objects Γ every $\rho \in \Gamma$ is finitely supported (i.e., has a support which is finite).

The submonoid **Cb** of **Sb** contains those $\pi \in \mathbf{Sb}$ satisfying for $\pi x, \pi y \notin \mathbf{2}$:

$$\pi x = \pi y \to x = y$$

This condition is like the condition for morphisms in the cubical category $\square$. This entails the following lemma.

Lemma 2.3.3. *For all $f\colon I \to J$ in $\square$ there exists a $\pi \in \mathbf{Cb}$ for which $\pi x = fx$ for all $x \in I$.*

Proof. See [71]. □

Theorem 2.3.4. *The category of finitely supported* **Cb***-sets* $[\mathbf{Cb}, \mathbf{Set}]_{\mathsf{fs}}$ *is equivalent to the category of cubical sets* **cSet**.

Proof. We only define the functor $F\colon \mathbf{cSet} \to [\mathbf{Cb}, \mathbf{Set}]_{\mathsf{fs}}$ and its inverse G on objects; for the detailed proof we refer the reader to [71]. For a cubical set X we take the colimit of X in $\square$ restricted to inclusions $I \subseteq J$, and define $FX = \varinjlim_I X(I)$. So an element in FX is an equivalence class $[I, u]$ with I a finite set of atoms and $u \in X(I)$; two such equivalence classes $[I, u]$ and $[J, v]$ are equal iff for some $K \supseteq I \cup J$, $u\iota = v\iota'$ where ι and ι' are the inclusions $I \subseteq K$ and $J \subseteq K$, respectively. For $\pi \in \mathbf{Cb}$ we have that $\pi|_I\colon I \to \pi(I) - \mathbf{2}$ is a morphism in $\square$, and we define $[I, u]\,\pi = [\pi(I) - \mathbf{2}, u(\pi|_I)]$.

Conversely, given a finitely supported **Cb**-set Y, GY is defined by

$$(GY)(I) = \{u \in Y \mid u \text{ is supported by } I\}.$$

For $u \in (GY)(I)$ and $f\colon I \to J$, $uf := u\,\pi$ with π as in Lemma 2.3.3; this doesn't depend on the choice of π since I supports u, and is well defined since $u\pi$ is supported by $\pi(I) - \mathbf{2} \subseteq J$ (cf. [71, Corollary 2.5]). □

Let Γ be a finitely supported **Cb**-set and $u \in \Gamma$; then u there is a least set of names supporting u denoted by $\mathrm{supp}(u)$ (cf. [71, Definition 2.6]). Then a set of atoms I supports u iff $\mathrm{supp}(u) \subseteq I$. We write $u \# v$ for if $\mathrm{supp}(u) \cap \mathrm{supp}(v) = \emptyset$ for $u, v \in \Gamma$. The set $\mathbb{A} \cup \mathbf{2}$ becomes a finitely supported **Cb**-set via $x\pi = \pi(x)$, $0\pi = 0$, and $1\pi = 1$, whose corresponding cubical set is isomorphic to the interval $\mathbb{I}$; clearly $\mathrm{supp}(x) = \{x\}$ and $\mathrm{supp}(0) = \mathrm{supp}(1) = \emptyset$.

Another way to present finitely supported **Cb**-sets is as nominal sets with extra structure.

Definition 2.3.5. The group $\mathrm{Per}(\mathbb{A})$ is given by the permutations in **Sb**, i.e., bijections $\pi\colon \mathbb{A} \to \mathbb{A}$ such that $\{x \in \mathbb{A} \mid \pi x \neq x\}$ is finite. The category of *nominal sets* **Nom** is the category of finitely supported $\mathrm{Per}(\mathbb{A})$-sets.

Each finitely supported **Cb**-set is also a nominal set. The notion of support w.r.t. the **Cb**-set structure coincides with the notion of support of nominal sets.

Definition 2.3.6. Let Γ be a nominal set. A structure of 01-*substitutions* on Γ is given by operations $(\!(x = c)\!)\colon \Gamma \to \Gamma$ for each name x and $c \in \mathbf{2}$ satisfying for $u \in \Gamma$:

(a) $(u(\!(x = c)\!))\pi = u\pi(\!(\pi x = c)\!)$;

(b) $u(\!(x = c)\!) \# x$;

(c) $u \# x \to u(\!(x = c)\!) = u$;

(d) $x \neq y \to u(\!(x = c)\!)(\!(y = d)\!) = u(\!(y = d)\!)(\!(x = c)\!)$.

The nominal sets with 01-substitutions constitute the object of the category **01Nom** whose morphisms are morphisms $\sigma\colon \Gamma \to \Delta$ in **Nom** such that $(\sigma(u))(\!(x = c)\!) = \sigma(u(\!(x = c)\!))$.

Lemma 2.3.7. *The categories* **01Nom** *and finitely supported* **Cb***-sets are equivalent. And thus the former is also equivalent to* **cSet***.*

Proof. See [71]. The main idea is to use that any element in **Cb** is a composition of swaps $(x\ y)$ and $(x = a)$. The latter are taken care of by the structure of 01-substitutions $(\!(x = a)\!)$. □

2.4 Separated Products

The equivalence of cubical sets with nominal sets equipped with 01-substitutions lets us translate important constructions on nominal sets to cubical sets. Let X and Y be cubical sets and $u \in X(I)$; recall that we wrote $x \# u$ if u degenerate along $x \in I$. If also $v \in Y(I)$, write $u \# v$ if there are $u' \in X(J)$ and $v' \in X(K)$ for $J, K \subseteq I$ with $J \cap K = \emptyset$ such that $u = u'\iota$ and $v = v'\iota'$ with ι and ι' being the respective inclusions $J \subseteq I$ and $K \subseteq I$. In that case $uf \# vf$ for $f\colon I \to I'$ witnessed by $u'(f|J)$ and $v'(f|K)$ and $f(J) \cap f(K) \subseteq \mathbf{2}$ since f is injective on names.

The *separated product* $X * Y$ of X and Y is given by

$$(X * Y)(I) = \{(u, v) \in X(I) \times Y(I) \mid u \# v\} \subseteq (X \times Y)(I)$$

and componentwise restrictions, making it a sub-cubical set of $X \times Y$.

It is easily verified that for J, K disjoint, we have

$$\mathbf{y}J * \mathbf{y}K \cong \mathbf{y}(J \cup K).$$

Moreover, $- * Y$ extends to a functor which has a right adjoint $Y \multimap -$, which we have already seen in Definition 2.2.7 as $[\mathbb{I}]-$ in the special case of $Y = \mathbb{I}$.

The functor on cubical sets $Y \multimap -$ is given by

$$(Y \multimap Z)(I) = \mathbf{cSet}(Y * \mathbf{y}I, Z)$$

for Z in **cSet** (this is natural in I and Z, and thus induces a endo-functor on **cSet**). That this is adjoint to $- * Y$ follows from the fact that $- * Y$ commutes with colimits.

Let us sketch that $[\mathbb{I}]X$ is isomorphic to $\mathbb{I} \multimap X$. Define the map $\varphi\colon [\mathbb{I}]X \to (\mathbb{I} \multimap X)$ as follows: let $(a, f) \in (\mathbb{I} * \mathbf{y}I)(J)$, i.e., $a \in \mathbb{I}(J)$ and $f\colon I \to J$ with

$a \# f$; the latter yields that $f = f'\iota$ with $f' \colon I \to J - a$ and ι the inclusion $I - a \subseteq I$. We define for $\langle x \rangle \omega$ in $([\mathbb{I}]X)(J)$

$$\varphi(\langle x \rangle \omega)(a, f) = \omega(f', x = a) \in X(J).$$

One can check that $\varphi(\langle x \rangle \omega)$ is an element in $(\mathbb{I} \multimap X)(J)$, and that φ is a morphism. The inverse $\psi \colon (\mathbb{I} \multimap X) \to [\mathbb{I}]X$ is given for $\theta \in (\mathbb{I} \multimap X)(I)$ by

$$\psi\theta = \langle x \rangle \theta(x, s_x)$$

for x fresh, so $x \# s_x$. One can check that this defines a morphism, which is indeed an inverse of φ.

Chapter 3

Kan Cubical Sets

As we have seen in the last chapter it is not possible to justify the elimination rules for the identity types defined from path spaces in the cubical set model. In this chapter we will introduce the notion of when a type $\Gamma \vdash A$ is a *uniform Kan type*; restricting to types with this condition is sufficient to justify the elimination rule for identity types defined from path spaces. The main work is to show that this notion is closed under the type formers.

3.1 The Uniform Kan Condition

The uniform Kan condition is about requiring fillers of "open box" like shapes. It is reminiscent of Daniel Kan's original extension axiom in [53]. Kan introduced this notion in order to give a combinatorial definition of homotopy groups. To simplify the discussion of the filling condition we will first only introduce the non-relative case. Let us first introduce these open box shapes and operations on them; these shapes correspond to *horns* in simplicial sets.

Definition 3.1.1 (Open Boxes)**.** Let I be a finite sets of names, $x, J \subseteq I$ with $x \notin J$ and $a \in \{0,1\}$. The triple $S = ((x,a); J; I)$ is called an *open box shape* on I; its *indices* $\langle S \rangle$ are given by

$$\langle S \rangle = \{(y,c) \mid c \in \mathbf{2}, y \in J, x \text{ and } (y,c) \neq (x,a)\} = \{(x,\bar{a})\} \cup J \times \mathbf{2}.$$

If $a = 1$, we call S a $+$-shape; otherwise, i.e., $a = 0$, a $-$-shape.

Let X be a cubical set. An *S-open box* in X (or simply *open box*) is given by a family $\vec{u}$ indexed by $\langle S \rangle$ such that $u_{yb} \in X(I - y)$ for $(y,b) \in \langle S \rangle$ and such that $\vec{u}$ is *adjacent compatible*, i.e., for all $(y,b), (z,c) \in \langle S \rangle$ with $y \neq z$ we have

$$u_{yb}(z = c) = u_{zc}(y = b) \tag{3.1}$$

The element $u_{x\bar{a}}$ of an S-open box $\vec{u}$ is called the *principal side* of $\vec{u}$ and x its *principal direction*; the sides u_{yc}, for $y \in J$, are called its *non-principal sides*,

and J its *non-principal directions*. We assume that the first entry v in $\vec{u} = v, \vec{v}$ is its principal side.

For $f\colon I \to K$ with $x, J \subseteq \mathrm{def}(f)$ and an S-open box $\vec{u}$ in I, we define the open box $\vec{u}f$ given by the components $u_{yb}(f - y) \in X(K - fy)$; this gives in an Sf-open box where $Sf = ((fx, a); fJ; K)$ for $S = ((x, a); J; I)$ and where fJ denotes the image of J under f.

Definition 3.1.2 (Kan Cubical Set)**.** A *uniform Kan cubical set* X, or simply *Kan cubical set*, is a cubical set X equipped with the following *filling operations*. For each open box shape S and S-open box $\vec{u}$ in X, we require and element

$$[X]_S\vec{u} \in X(I)$$

such that for $(y, b) \in \langle S\rangle$

$$([X]_S\vec{u})(y = b) = u_{yb}$$

and additionally for each $f\colon I \to K$ with $x, J \subseteq \mathrm{def}(f)$ the following *uniformity condition* (or *coherence condition*):

$$([X]_S\vec{u})f = [X]_{Sf}(\vec{u}f) \tag{3.2}$$

If S is a $+$-shape, we denote $[X]_S\vec{u}$ by $X{\uparrow_S}\vec{u}$; and if S is a $-$-shape by $X{\downarrow_S}\vec{u}$. Moreover, we usually suppress the shape of the box and tacitly assume that an open box fits the filling operator. We also give names to the face in the principal direction

$$X^+\vec{u} = (X{\uparrow}\vec{u})(x = 1) \qquad \text{and} \qquad X^-\vec{u} = (X{\downarrow}\vec{u})(x = 0)$$

and call them the *induced composition operations*.

It is also possible to consider a cubical set X with only *composition operations*: for each open box shape $S = ((x, a); J; I)$ and S-open box we require $|X|_S\vec{u} \in X(I - x)$ such that for $(y, b) \in \langle S\rangle$, $(|X|_S\vec{u})(y = b) = u_{yb}(x = a)$ and for $f\colon I - x \to J$ defined on $I - x$, we require the uniformity conditions

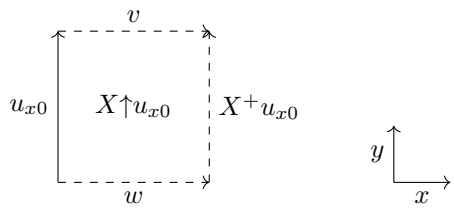
$$(|X|_S\vec{u})f = |X|_{Sf'}(\vec{u}f')$$

with $f' = (f, x = z)\colon I \to J, z$ for z fresh w.r.t. J. (One should consider the name x in $|X|_S\vec{u} \in X$ to be *bound*.) The induced composition operations of a Kan cubical set are composition operations in this sense.

We emphasize that the above definition requires a *fixed choice* of fillers of open boxes and is thus equipped with "algebraic" operations, and this algebraic presentation is crucial in order to formulate the uniformity condition (3.2). A similar notion for simplicial sets is that of an algebraic Kan complex [65]. But note that these come without additional equations like the uniformity condition above. Similar operations for semi-simplicial sets have been considered in [11].

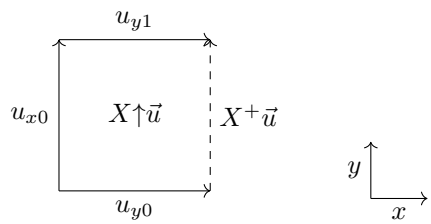
We will sometimes refer to the operations of a Kan cubical set as its *Kan structure*.

Let us analyze the filling operations of a uniform Kan cubical set X in low dimensions. The simplest case is where J is empty (with the same notations as in the definition above): a corresponding open box, for say $a = 1$, is simply given by an element $u_{x0} \in X(I - x)$; the filler $X{\uparrow}u_{x0}$ is an element in $X(I)$ with $(X{\uparrow}u_{x0})(x = 0) = u_{x0}$. Thus if say $y \in I$, this gives a square:

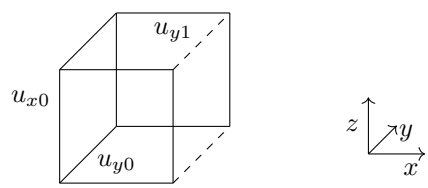


By definition the right hand face is the composition $X^+u_{x0} \in X(I - x)$. But what about the top and bottom faces v and w? The uniformity conditions guarantee that the filling operation commutes with the face operations $(y = 1)$ and $(y = 0)$, and thus we know that $v = X{\uparrow}(u_{x0}(y = 1))$ and $u = X{\uparrow}(u_{x0}(y = 0))$. Moreover, if u_{x0} happens to be degenerate along y, the uniformity condition entails that $X{\uparrow}u_{x0} = vs_y$.

Let us now assume that $J = y$; then an open box of the corresponding shape comes with three elements: $\vec{u} = u_{x0}, u_{y0}, u_{y1}$ such that $u_{x0} \in X(I - x)$ and $u_{y0}, u_{y1} \in X(I - y)$, and that $u_{x0}(y = b) = u_{yb}(x = 0)$ for $b \in \mathbf{2}$. Thus the situation can be depicted as a "U-shape" whose filler $X{\uparrow}\vec{u} \in X(I)$ is as indicated:



If we also have another $z \in I$ this situation also can be depicted as follows, where the filler $X{\uparrow}\vec{u}$ is the whole cube (omitting the arrow tips):



The uniformity conditions ensure that the top face $(z = 1)$ of the cube is the filler of the top "U-shape" $\vec{u}(z = 1)$; and likewise for the bottom. And similarly, if *all* of the sides in $\vec{u}$ are degenerate in direction z, the filling cube is the degenerate of the filling square of the top "U-shape" (which is in this case equal to the lower one).

And so on: with J having two elements, corresponding open boxes become cubes with a missing side and missing interior etc.

(But note that the uniformity conditions say nothing about how two filling operations for different cardinalities of the J parameter relate.)

Lemma 3.1.3. *Let R be a commutative ring. The cubical set $P = R[\cdot]$ induced by R as defined in Example 2.2.3 is a Kan cubical set.*

Proof. Let $S = ((x, a); J; I)$ and $\vec{p}$ an open box of shape S in P. We will construct the filler $p = [P]_S\vec{p}$ by an iterated linear interpolation. First we define $p_J \in R[I]$ such that $p_J(y = b) = p_{yb}$ for $y \in J$ by induction on the size of J: we start with $p_\emptyset = 0$; if $J = z, K$, we set

$$p_{z,K} = p_K + (1 - z)(p_{z0} - p_K(z = 0)) + z(p_{z1} - p_K(z = 1)).$$

Note that if $K = \emptyset$, this p_z is simply the linear interpolation

$$p_z = (1 - z)p_{z0} + zp_{z1}.$$

We have for $b \in \mathbf{2}$, $p_{z,K}(z = b) = p_K(z = b) + p_{zb} - p_K(z = b) = p_{zb}$ and for $y \in K$ with the IH and the fact that $\vec{p}$ is adjacent compatible:

$$\begin{aligned} p_{z,K}(y = b) &= p_K(y = b) + (1 - z)\big(p_{z0}(y = b) - p_K(y = b)(z = 0)\big) \\ &\quad + z\big(p_{z1}(y = b) - p_K(y = b)(z = 1)\big) \\ &= p_{yb} + (1 - z)\big(p_{yb}(z = 0) - p_{yb}(z = 0)\big) \\ &\quad + z(p_{yb}(z = 1) - p_{yb}(z = 1)) = p_{yb} \end{aligned}$$

As a last step we define the filler $p \in R[I]$ in case $a = 1$ by

$$p = p_J + (1 - x)(p_{x0} - p_J(x = 0))$$

and analogously for $a = 0$. This has the correct faces and the definition is independent on the order we chose to pick the names of J as is readily checked. The definition satisfies the uniformity conditions: observe that $(p_J)f$ for f defined on J is the same as the corresponding polynomial for the $p_{yb}(f-y)$ with $y \in J$; similarly, this extends to pf for f defined on x, J. □

Lemma 3.1.4. *The cubical nerve $\mathrm{N}(\mathcal{G})$ of a (small) groupoid $\mathcal{G}$ is a Kan cubical set.*

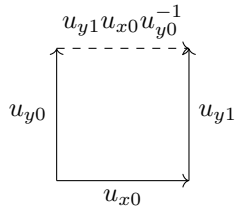
Proof. Let $S = ((x, 1); J; I)$ and $\vec{u}$ an S-open box in $\mathrm{N}(\mathcal{G})$. The proof for $-$-open boxes is similar. In case $J = \emptyset$, we define the filling of $\vec{u}$ (which only consists of u_{x0}) by us_x. This clearly satisfies the required uniformity conditions. In case J contains at least two elements, we argue that the input box $\vec{u}$ already contains all the needed edges and can be uniquely considered as an I-cube: we define the filler $u\colon \mathbf{2}^I \to G$ on an object $\alpha\colon I \to \mathbf{2}$ by $u\alpha = u_{yb}(\alpha - y)$ for $\alpha y = b$ with $(y, b) \in \langle S\rangle$. Note that there has to exist such a (y, b) since $J \neq \emptyset$. Also, this is well defined since $\vec{u}$ is adjacent compatible.

On morphisms $\alpha \leq \beta$ we first define $u(\alpha \leq \beta)$ as follows. If both $\alpha y = \beta y = b$ for some $(y, b) \in \langle S \rangle$, then we take $u_{yb}(\alpha - y \leq \beta - y)$. Otherwise, there are $(y, b), (z, c) \in \langle S \rangle$ with $\alpha y = b$ and $\beta z = c$ with $y \neq z$ since J contains at least two elements; then we take

$$u_{zc}((\beta - z, x = 0) \leq \beta - z) \circ u_{x0}(\alpha - x \leq \beta - x) \circ (u_{yb}((\alpha - y, x = 0) \leq \alpha - y))^{-1}$$

which is forced by the groupoid structure. This determines u uniquely from the fact that u has to have $\vec{u}$ as corresponding faces.

It remains the case where J consists exactly of one element, say y. We construct the filler by induction on $I - (x, J)$ together with showing that the filler is unique, and hence satisfies the required uniformity conditions. In case $I - (x, J)$ is empty, we construct the composition as follows:



This also determines the filler $u \in (\mathrm{N}(\mathcal{G}))(x, y)$ since the diagram commutes, and is unique by the groupoid structure. Now in case, $I - (x, J)$ contains z, we inductively fill $\vec{u}(z = 0)$ and $\vec{u}(z = 1)$ to unique u_{z0} and u_{z1} in $(\mathrm{N}(\mathcal{G}))(I - z)$, respectively. Now we define the filler $u \in (\mathrm{N}(\mathcal{G}))(I)$ of $\vec{u}$ as the filler of the extended open box $\vec{u}, u_{z0}, u_{z1}$ (which we already constructed above). If $u' \in (\mathrm{N}(\mathcal{G}))(I)$ is another filler of $\vec{u}$, then by IH, $u'(z = b)$ has to be equal to u_{zb}, and hence to $u(z = b)$. But then u and u' are both also fillers of $\vec{u}, u_{z0}, u_{z1}$ which we have shown to be unique above. □

Definition 3.1.5. Let Γ be a cubical set. A type $\Gamma \vdash A$ is a *uniform Kan type*, or simply *Kan type*, if it is equipped with the following operations. Let $\alpha \in \Gamma(I)$, S an open box shape on I, and $\vec{u}$ an S-open box in $A\alpha$, i.e., $\vec{u}$ is an $\langle S \rangle$-indexed family where the component u_{yb} for $(y, b) \in \langle S \rangle$ is an element $u_{yb} \in A\alpha(y = b)$, such that $\vec{u}$ is adjacent compatible. We require fillers

$$[A\alpha]_S \vec{u} \in A\alpha$$

such that for $(y, b) \in \langle S \rangle$

$$([A\alpha]_S \vec{u})(z = b) = u_{yb}$$

and additionally for each $f \colon I \to K$ with $x, J \subseteq \mathrm{def}(f)$ the *uniformity condition* holds, i.e.,

$$([A\alpha]_S \vec{u})f = [A\alpha]_{Sf}(\vec{u}f).$$

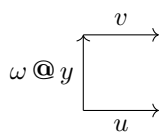
We will use the analogous notation $A\alpha{\uparrow}\vec{u}, A\alpha{\downarrow}\vec{u}, A\alpha^{+}\vec{u}, A\alpha^{-}\vec{u}$ as in Definition 3.1.2.

Remark 3.1.6. To get the definition of when a map of cubical sets $\sigma\colon \Delta \to \Gamma$ is a (uniform) Kan fibration replace $A\alpha$ by $\sigma^{-1}(\alpha)$ in the above definition. Then $\Gamma \vdash A$ is a uniform Kan type iff the projection $\mathtt{p}\colon \Gamma.A \to \Gamma$ is a uniform Kan fibration.

As an immediate consequence of the definition we get:

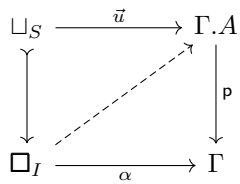
Lemma 3.1.7. *A X cubical set is a Kan cubical set if and only if, X considered as a type in the empty context $1 \vdash X$ is a Kan type.*

Remark 3.1.8. The open box shapes with non-principal faces appear naturally when we want that the operation $u \mapsto A\rho\uparrow u$ extends to the identity type, say, $X \times X \vdash \mathtt{Id}_X$ for a cubical set X. Given $u, v \in X(I, x)$ and $\omega \in \mathtt{Id}_X(u_0, v_0)$ (where $u_0 = u(x = 0)$ etc.), so $\omega \mathbin{@} 0 = u_0$ and $\omega \mathbin{@} 1 = v_0$; let y be fresh. A filler $\mathtt{Id}_X(u, v)\uparrow\omega$ is a filler of the following open box shape (modulo the abstraction $\langle y\rangle -$):



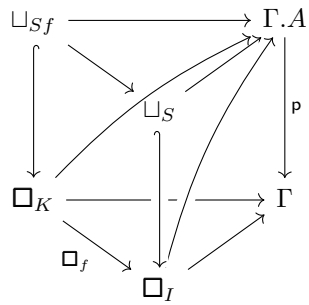
Thus it is natural to require these more general filling operations on X.

Remark 3.1.9. Similar to the lifting condition for simplicial sets w.r.t. horn inclusions, we can formulate the uniform Kan condition of a type $\Gamma \vdash A$ as the lifting of maps. Let $\sqcup_S(K) \subseteq \square_I(K)$ for $S = ((x, a); J; I)$ consist of those $f\colon I \to K$ such that $fy = b$ for some $(y, b) \in \langle S\rangle$. This defines a sub-cubical set of $\square_I$. Open boxes of shape S in a cubical set Γ correspond to morphisms $\sqcup_S \to \Gamma$. The existence of fillers are *chosen* diagonal fillers for every outer square (where the lower horizontal map corresponds to an I-cube in Γ, and the upper horizontal map to an open box):



For $f\colon I \to K$ defined on all of x, J the map (given by the Yoneda embedding) $\square_f\colon \square_K \to \square_I$ restricts to a map $\sqcup_{Sf} \to \sqcup_S$ (since for $g \in \sqcup_{Sf}(L)$, i.e., $gz = b$ for $(z, b) \in \langle Sf\rangle$, we have $z = fy$ for some $y \in J$, and thus $fg = \square_f g \in \sqcup_S(L)$). The uniformity conditions translate to the commutation of the following prism

where the two diagonal (slightly bent) arrows into $\Gamma.A$ are the chosen fillers:



Remark 3.1.10. Note that for $S = ((x, a); J; I)$ with $I = x, J, K$, where x, J and K disjoint, we have

$$\sqcup_S \cong \sqcup_{((x,a);J;J)} * \square_K. \tag{3.3}$$

Moreover, for $f\colon I \to I'$ defined on x, J, we can write $I' = fx, fJ, K'$ (disjoint). We have the induced map

$$\sqcup_{((x,a);J;J)} * \square_K \cong \sqcup_S \longrightarrow \sqcup_{Sf} \cong \sqcup_{((fx,a);fJ;fJ)} * \square_{K'}$$

whose right component is induced by the renaming on x, J and whose left component is induced by the morphism $f - x, J\colon K \to K'$.

Now this suggests yet another reformulation of the Kan structure on a cubical set due to Peter Lumsdaine of which we only give a short sketch. Let $\sqcup^a_{x;J} := \sqcup_{((x,a);J;J)} \subseteq \square_{x,J}$ and consider the cubical set $\sqcup^a_{x;J} \multimap X$ for a cubical set X: by the Yoneda Lemma, its K-cubes are

$$\sqcup^a_{x;J} * \square_K \to X$$

which in case that x, J and K are disjoint corresponds to (by (3.3))

$$\sqcup_{((x,a);J;x,J,K)} \to X,$$

i.e., the $((x, a); J; x, J, K)$ open boxes in X. Consider the canonical map

$$r^a_{x;J}\colon (\square_{x,J} \multimap X) \to (\sqcup^a_{x;J} \multimap X)$$

induced by the inclusion $\sqcup^a_{x;J} \subseteq \square_{x,J}$.

Assume now that X has a Kan structure. We define a section $s^a_{x;J}$ of $r^a_{x;J}$. This amounts to give level-wise sections

$$(\sqcup^a_{x;J} \multimap X)(K) \to (\square_{x,J} \multimap X)(K)$$

natural in K. But the left hand side corresponds to open boxes $\vec{u}$ of shape $((x, a); J; x, J, K)$, and the right hand side corresponds to x, J, K cubes, and thus we can use $[X]_S\vec{u}$ to define the image of $\vec{u}$. Now this is natural in K

since we have the uniformity conditions for maps $f\colon x, J, K \to x, J, K'$ which are the identity on x, J. The uniformity conditions for renaming in x, J yield additional equations on the sections: a renaming $f\colon x, J \to fx, fJ$ defined on all of x, J induces the vertical maps in

$$\begin{array}{ccc} (\square_{x,J} \multimap X) & \rightleftarrows & (\sqcup^a_{x;J} \multimap X) \\ \downarrow & & \downarrow \\ (\square_{fx,fJ} \multimap X) & \rightleftarrows & (\sqcup^a_{fx;fJ} \multimap X) \end{array} \tag{3.4}$$

where the horizontal maps are given by the corresponding sections and retractions. Now the (full) uniformity conditions yield that the diagram commutes. The converse is also true: if we have a choice of sections for all the $r^a_{x;J}$ such that all the squares of the form (3.4) commute, then X has a Kan structure.

3.2 The Kan Cubical Set Model

In this section we show that Kan types are closed under the type formers and gives rise to a CwF supporting Σ-, and Π-types. It is crucial to observe that the filling operations are part of the definition when $\Gamma \vdash A$ is a Kan type. That means that for Kan types $\Gamma \vdash A$ and $\Gamma \vdash B$ we can have $A = B$ as cubical sets but *not* necessarily as Kan types, i.e., their Kan structures might not coincide. Thus we have to check *coherence conditions*, i.e., we have to verify that the CwF equations between types are preserved by the filling operations.

Theorem 3.2.1. *Kan types give rise to a CwF supporting* Π- *and* Σ*-types, where*

1. *contexts* $\Gamma \vdash$ *are interpreted by cubical sets;*
2. *substitutions* $\sigma\colon \Delta \to \Gamma$ *are maps of cubical sets;*
3. *types* $\Gamma \vdash A$ *are Kan types;*
4. *terms of a Kan type* $\Gamma \vdash A$ *are terms of* $\Gamma \vdash A$ *considered as a type in the cubical set (i.e., presheaf) sense.*

Note that by Section 1.2 and the fact that cubical sets are just presheaves on the cubical category $\square$, we get that cubical sets induce a CwF. The difference to the model we give in the theorem is in item 3, that is, types are equipped with a Kan structure. The proof of Theorem 3.2.1 spans the rest of this section and the definition of the respective Kan structures are given in the proofs of the following theorems. The part of the CwF which is shared with the CwF of cubical sets is given in the same manner. So context extension and the definition of the required terms and context morphisms are defined to be the same as for the cubical set model. Also, the constructions on types are the same except that we have to care about the Kan structure as well. In other

words, forgetting the Kan structure induces a morphism of CwFs preserving Π and Σ.

Let us also mention (without proof) that the model also supports base types like the natural numbers. Their interpretation is given via the constant presheaf.

We will reserve "type" for a type in the cubical set sense and use "Kan type" for a type with its Kan structure.

Theorem 3.2.2. *If $\Gamma \vdash A$ is a Kan type, $\Delta \vdash$, and $\sigma\colon \Delta \to \Gamma$ a context morphism, then also the type $\Delta \vdash A\sigma$ is a Kan type. Moreover, the Kan structure is such that we get:*

$$A\mathbf{1} = A \qquad (A\sigma)\tau = A(\sigma\tau)$$

Proof. For an I-cube α of Δ recall that we defined $(A\sigma)\alpha = A(\sigma\alpha)$. We define the filling operations of $(A\sigma)\alpha$ to be those of $A(\sigma\alpha)$, i.e., we set $[(A\sigma)\alpha]_S\vec{u} = [A(\sigma\alpha)]_S\vec{u}$ for an open box $\vec{u}$. With this definition it is clear that A and $A\mathbf{1}$ have the same Kan structure, and likewise for the other equation. □

Theorem 3.2.3. *If $\Gamma \vdash A$ and $\Gamma.A \vdash B$ are Kan types, then so is the type $\Gamma \vdash \Sigma AB$. Moreover, $(\Sigma AB)\sigma = \Sigma(A\sigma)(B(\sigma\mathsf{p}, \mathsf{q}))$ as Kan types.*

Proof. Let $\vec{u}$ be an S-open box in $(\Sigma AB)\alpha$ for $\alpha \in \Gamma(I)$. Then with $w_{yb} = (u_{yb}, v_{yb})$ where $u_{yb} \in A\alpha(y = b)$ and $v_{yb} \in B(\alpha(y = b), u_{yb})$ for $(y, b) \in \langle S\rangle$, we get that $\vec{u}$ is an S-open box in $A\alpha$ which we can fill to $u = [A\alpha]_S\vec{u}$. Now $\vec{v}$ is also a S-open box in $B(\alpha, u)$ and we set

$$[(\Sigma AB)\alpha]_S\vec{w} = (u, [B(\alpha, u)]_S\vec{v}).$$

This definitions satisfies the required uniformity conditions as they are satisfied for $\Gamma \vdash A$ and $\Gamma.A \vdash B$.

Now if $\alpha = \sigma\beta$ for $\sigma\colon \Delta \to \Gamma$, we get that $u = [(A\sigma)\beta]_S\vec{u}$ and

$$[B(\sigma\beta, u)]_S\vec{v} = [(B(\sigma\mathsf{p}, \mathsf{q}))(\beta, u)]_S\vec{v},$$

yielding $(\Sigma AB)\sigma = \Sigma(A\sigma)(B(\sigma\mathsf{p}, \mathsf{q}))$ as Kan types. □

For the next theorem we need some notations. Given $\Gamma \vdash A$, $\alpha \in \Gamma(I)$, $u \in A\alpha$, and a shape $S = ((x, a); J; I)$, we define an S-open box u^S by "carving" out an S-shape from u, i.e., u^S is given by the components $u^S_{yc} = u(y = c) \in A\alpha(y = c)$. Note that the filling operation $[A\alpha]_S$ is a section of this operation $(-)^S$. Moreover, for $f\colon I \to J$ defined on J, x we have $(u^S)f = u^{Sf}$.

For $\Gamma \vdash \Pi AB$ and S-shapes $\vec{w}$ in $(\Pi AB)\alpha$ and $\vec{u}$ in $A\alpha$ such that $u^S = \vec{u}$ for some $u \in A\alpha$, we define the S-shape $\vec{w}\vec{u}$ in $B(\alpha, u)$ by the components $(\vec{w}\vec{u})_{yc} = (w_{yc})_{\mathbf{1}}u_{yc} \in B(\alpha(y = c), u(y = c))$. (Recall from Section 1.2.1 that elements in Π-types are *families* of dependent functions.) For $f\colon I \to J$ defined on J, x this satisfies $(\vec{w}\vec{u})f = (\vec{w}f)(\vec{v}f)$ since:

$$\begin{aligned}(\vec{w}\vec{u})_{yc}(f - y) &= ((w_{yc})_{\mathbf{1}}u_{yc})(f - y)\\ &= (w_{yc})_{(f-y)}(u_{yc}(f - y))\\ &= (w_{yc}(f - y))_{\mathbf{1}}(u_{yc}(f - y))\end{aligned}$$

Theorem 3.2.4. *If $\Gamma \vdash A$ and $\Gamma.A \vdash B$ are Kan types, then so is $\Gamma \vdash \Pi AB$. Moreover, $(\Pi AB)\sigma = \Pi(A\sigma)(B(\sigma \mathsf{p}, \mathsf{q}))$ as Kan types.*

Proof. Let $C = \Pi AB$ and let S be an open box shape. We assume that S is a $+$-shape, i.e., of the form $S = ((x,1); J; I)$; the case of $-$-shapes is symmetric.

First, we will define the composition operations $C\alpha^+_S\vec{w} \in (\Pi AB)\alpha(x=1)$ for $\alpha \in \Gamma(I)$ and $\vec{w}$ an S-open box in $(\Pi AB)\alpha$. This amounts to define a family of dependent functions $(C\alpha^+_S\vec{w})_f$ in $\prod_{u \in A\alpha(x=1)_f} B(\alpha(x=1)f, u)$ for all $f\colon I - x \to K$, such that

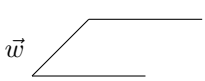
$$\big((C\alpha^+_S\vec{w})_f(u)\big)g = (C\alpha^+_S\vec{w})_{fg}(ug). \tag{3.5}$$

We will first define $(C\alpha^+_S w)_f$ for $f = \mathbf{1}\colon I - x \to I - x$. For this let $u \in A\alpha(x=1)$. We use the Kan fillings with shape $S_x = ((x,0); \emptyset; I)$ to extend u to $A\alpha\downarrow_x u \in A\alpha$ (with $\downarrow_x$ for $\downarrow_{S_x}$), of which we carve out an S-shape we then apply to $\vec{w}$, and map the result up:

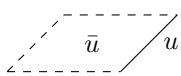
$$(C\alpha^+_S\vec{w})_{\mathbf{1}}(u) = B(\alpha, A\alpha\downarrow_x u)^+_S(\vec{w}\,(A\alpha\downarrow_x u)^S) \tag{3.6}$$

which is in $B(\alpha(x=1), u)$ as $(A\alpha\downarrow_x u)(x=1) = u$. This defines $(C\alpha^+_S\vec{w})_{\mathbf{1}}$ for arbitrary α and w. Note that to give the open box $(A\alpha\downarrow_x u)^S$ we only need composition operations of $\Gamma \vdash A$ (not so in the argument to B).

Let us illustrate this (where we assume one non-principal direction y). We are given $\vec{w}$

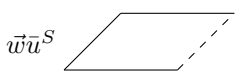


and we are given u which we fill to $\bar{u} = A\alpha\downarrow_x u$



in $A\alpha$

of which we carve out the open box $\bar{u}^S$ and apply $\vec{w}$, which we then fill in $B(\alpha, \bar{u})$



(3.7)

where the last dashed line on the right is the definition of $(C\alpha^+_S\vec{w})_{\mathbf{1}}$.

For general $f\colon I - x \to K$ we define $(C\alpha^+_S\vec{w})_f$ as follows. In case there is $(y,c) \in \langle S\rangle$ such that $fy = c$ we write f as $f = (y=c)(f-y)$ with $(f-y)\colon I - x, y \to K$. (Note that $y \neq x$.) We define

$$(C\alpha^+_S\vec{w})_f = (w_{yc})_{(x=1)(f-y)}. \tag{3.8}$$

This is well defined since $\vec{w}$ is adjacent compatible. Note that this ensures

$$(C\alpha^+_S\vec{w})(y=c) = w_{yc}(x=1).$$

Otherwise, i.e., f is defined on J we let z be fresh w.r.t. K (e.g., take $z = x_K$) and set

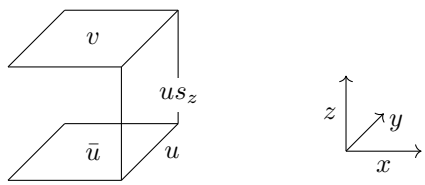
$$(C\alpha^+_S\vec{w})_f = \big((C\alpha f^z_x)^+_{Sf^z_x}(\vec{w}f^z_x)\big)_{\mathbf{1}} \tag{3.9}$$

where f^z_x is $(f, x = z)\colon I \to K, z$. By the uniformity conditions, this definition does not depend on the choice of z, and we also get by uniformity, (3.6), and the discussion above

$$\big((C\alpha^+_S\vec{w})_{\mathbf{1}}(u)\big)f = \big((C\alpha f^z_x)^+_{Sf^z_x}(\vec{w}f^z_x)\big)_{\mathbf{1}}(uf). \tag{3.10}$$

Note, that (3.9) says that the family $(C\alpha^+_S\vec{w})_f$ is determined by the value at $f = \mathbf{1}$ (with different α, S and $\vec{w}$). The uniformity condition follows from (3.9) (note that the left hand side is $((C\alpha^+_S\vec{w})f)_{\mathbf{1}}$ by definition). Equation (3.5) follows from (3.10) together with (3.8) and (3.9); more formally, to prove (3.5) one distinguishes cases on the definedness of f. If f is not defined on one of the non-principal sides, (3.5) follows from (3.8). Otherwise, the left hand side in (3.5) is given by (3.9), in which case one distinguishes cases on the definedness of g: in case g is not defined on one of the corresponding non-principal sides one uses (3.8) again, and otherwise, uses (3.10). Thus we obtain an element in $C\alpha(x = 1)$.

Next we define $C\alpha\uparrow_S\vec{w} \in C\alpha$; we do so again by first defining $(C\alpha\uparrow_S\vec{w})_f$ for $f = \mathbf{1}\colon I \to I$. Let $v \in A\alpha$, $u = v(x = 1)$, and let z be fresh (e.g., $z = x_I$). Consider the shape $S_{x,z} = ((x, 0); z; I)$; we get an $S_{x,z}$ open box us_z, $A\alpha\downarrow_x u$, v in $A\alpha s_z$ (where us_z is the principal side and the next two sides are at $(z, 0)$ and $(z, 1)$ respectively), illustrated as (again only with one non-principal side y, $\bar{u} = A\alpha\downarrow_x u$, and all sides should be considered "solid")



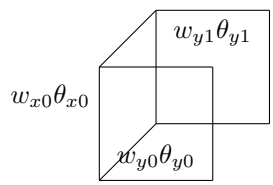
which we fill to

$$\theta = A\alpha s_z\downarrow_{x,z}(us_z, A\alpha\downarrow_x u, v) \in A\alpha s_z$$

where we wrote $\downarrow_{x,z}$ for $\downarrow_{S_{x,z}}$. From this we carve out an $Ss_z = ((x, 1); J; I, z)$ open box and apply it to $\vec{w}s_z$ to get an Ss_z open box

$$(\vec{w}s_z)\,\theta^{Ss_z} \text{ in } B(\alpha s_z, \theta), \tag{3.11}$$

or as picture (where we write $w_{x0}\theta_{x0}$ for $(w_{x0})_{s_z}(\theta(x = 0))$ etc.)



Notice that if we take the face $(z = 0)$ in the above picture, we get the same open box $\vec{w}\bar{u}^S$ as in (3.7).

We define the open box χ in $B(\alpha s_z, \theta)$ of shape $((z,1); x, J; I, z)$ with the principal side

$$B(\alpha, A\alpha{\downarrow_S}u){\uparrow_S}(\vec{w}\,(A\alpha{\downarrow_x}u)^S)$$

(which is the square in (3.7)) and where the non-principal side at $(x,1)$ is $\big((C\alpha^+_S\vec{w})_{\mathbf{1}}(u)\big)s_z$; these are compatible by construction (3.6); the non-principal side at $(x,0)$ is given by the principal side of the open box (3.11); the non-principal sides in directions J are the non-principal sides of (3.11).

We fill this to obtain the definition

$$(C\alpha{\uparrow}\vec{w})_{\mathbf{1}}(v) = B(\alpha s_z, \theta)^+\chi \tag{3.12}$$

yielding an element in $B(\alpha, v)$ since $\theta(z = 1) = v$ by definition of θ. This concludes the definition of $((C\alpha){\uparrow}\vec{w})_{\mathbf{1}}$ for all α and $\vec{w}$.

For general $f\colon I \to K$ we define $(C\alpha{\uparrow}\vec{w})_f$ by distinguishing cases:

$$(C\alpha{\uparrow_S}\vec{w})_f = \begin{cases} (C\alpha^+_S\vec{w})_{(f-x)} & \text{if } fx = 1, \\ (w_{yc})_{(f-y)} & \text{if } fy = c \text{ for some } (y,c) \in \langle S\rangle, \\ (C\alpha f{\uparrow_{Sfx}}\vec{w}f)_{\mathbf{1}} & \text{otherwise, i.e., } f \text{ is defined on } x, J. \end{cases}$$

where $(f - x)\colon I - x \to K$ and $(f - y)\colon I - y \to K$.

To conclude that this definition is a well-defined element of $C\alpha$ and satisfies the uniformity condition we need to verify that

$$\big((C\alpha{\uparrow}\vec{w})_{\mathbf{1}}(v)\big)f = (C\alpha{\uparrow}\vec{w})_f(vf) \tag{3.13}$$

for $f\colon I \to K$. If for some $y \in x, J$, fy is not defined, (3.13) follows from how the open box χ is defined. Otherwise, i.e., f is defined on all x, J, (3.13) follows by inspecting that, in the definition of $(C\alpha{\uparrow}\vec{w})_{\mathbf{1}}(v)$,

$$(B(\alpha s_z, \theta)^+\chi)f = B(\alpha f s_{z'}, \theta f')^+(\chi f')$$

where $f' = (f, z = z')$ and z' fresh w.r.t. K. And $\theta f'$ and $\chi f'$ are, by uniformity, exactly those arguments appearing in the definition of $(C\alpha f{\uparrow}\vec{w}f)_{\mathbf{1}}(vf)$ which is the right hand side of (3.13).

To verify that the Kan structure of $\Pi(A\sigma)(B(\sigma\mathsf{p}, \mathsf{q}))$ (as defined above) is equal to the Kan structure for $(\Pi\, A\, B)\sigma$ (as defined in the proof of Theorem 3.2.2), assume that above $\alpha = \sigma\beta$ for $\sigma\colon \Delta \to \Gamma$; then $C\alpha = ((\Pi\, A\, B)\sigma)\beta$ and in equation (3.6) we have

$$\begin{aligned} &B(\sigma\beta, A(\sigma\beta){\downarrow_x}u)^+(\vec{w}(A(\sigma\beta){\downarrow_x}u)^S) \\ &= (B(\sigma\mathsf{p}, \mathsf{q}))(\beta, (A\sigma)\beta{\downarrow_x}u)^+(\vec{w}((A\sigma)\beta{\downarrow_x}u)^S) \end{aligned}$$

and the right hand side is the definition of $\big(\Pi(A\sigma)(B(\sigma\mathsf{p}, \mathsf{q}))^+\vec{w}\big)_1(u)$. Similarly for the other parts of the definition. □

Remark 3.2.5. We also get a CwF if in Theorem 3.2.1, we require contexts to be *Kan* cubical sets instead of just cubical sets. The crucial point that this works is that if $\Gamma \vdash$ is a Kan cubical set and $\Gamma \vdash A$ is a Kan type, then $\Gamma.A \vdash$ is a Kan cubical set. The proof of this statement is along the lines of the closure of Kan types under Σ-types (see Theorem 3.2.3); in fact, it can be derived using that $1 \vdash \Gamma$ is a Kan type.

3.3 Identity Types

We will now define the (weak) identity type of a cubical set and justify an elimination operator and functional extensionality for it. The underlying idea of the definition of identity type is that a proof of equality $\mathtt{Id}_A(a,b)$ should be a path with endpoints a and b.

Recall the definition of the non-dependent path space (Definition 2.2.7), non-dependent identity types (Definition 2.2.8), and the notation used there.

Definition 3.3.1. Let $\Gamma \vdash A$, $\Gamma \vdash a : A$, $\Gamma \vdash b : A$. The *(weak) identity type* $\Gamma \vdash \mathtt{Id}_A(a,b)$ is defined as follows: for $\rho \in \Gamma(I)$ an element $\omega \in (\mathtt{Id}_A(a,b))\rho$ is an element $\omega \in A(\rho s_{x_I})$ such that $\omega(x_I = 0) = a\rho$ and $\omega(x_I = 1) = b\rho$. The restriction by a $f\colon I \to J$ of $\omega \in (\mathtt{Id}_A(a,b))\rho$ is defined, as for the non-dependent identity type, by $\omega f = \omega(f, x_I = x_J)$ (where on right we use the restriction of A).

As in 2.2.8 we could have used equivalence classes $\langle x\rangle p$ with $p \in A(\rho s_x)$ where $\rho \in \Gamma(I)$, x fresh for I, and $\langle x\rangle p = \langle y\rangle q$ if $p(x = y) = q$. The operation $\omega \mathbin{@} a$ for $a \in \mathbf{2}$ or a fresh is defined as there, i.e., $\omega \mathbin{@} a = \omega(x_I = a)$. For $p \in A\rho s_x$ we set $\langle x\rangle p = p(x_I = x)$. Note that for $f\colon I \to J$ and $\omega \in (\mathtt{Id}_A(a,b))\rho$, $x \notin I$ we have $(\omega \mathbin{@} x)(f, x = a) = \omega f \mathbin{@} a$. No matter which definition is used we have the notions of $\langle x\rangle p$ and $\omega \mathbin{@} a$ as in 2.2.8.

Theorem 3.3.2. *Kan types are closed under identity types, i.e., if $\Gamma \vdash A$ is a Kan type, $\Gamma \vdash a : A$, and $\Gamma \vdash b : A$, then $\Gamma \vdash \mathtt{Id}_A(a,b)$ is a Kan type. Moreover, for $\sigma\colon \Delta \to \Gamma$, $(\mathtt{Id}_A(a,b))\sigma = \mathtt{Id}_{A\sigma}(a\sigma, b\sigma)$ as Kan types.*

Proof. Let $\vec{\omega}$ be an S-open box in $(\mathtt{Id}_A(a,b))\rho$ with $\rho \in \Gamma(I)$ and let z be fresh; then $\vec{\omega} \mathbin{@} z$ (component-wise) is an S-open box in $A\rho s_z$. We extend this to an S, z-open box in $A\rho s_z$, where S, z is like S but with the non-principal side z added, given by $\vec{\omega} \mathbin{@} z, a\rho, b\rho$ and define

$$[\mathtt{Id}_A(a,b)\rho]_S\vec{\omega} = \langle z\rangle\, [A\rho s_z]_{S,z}(\vec{\omega} \mathbin{@} z, a\rho, b\rho) \tag{3.14}$$

Note that by the definition of the extended open box

$$([\mathtt{Id}_A(a,b)\rho]_S\vec{\omega}) \mathbin{@} 0 = a\rho \text{ and } ([\mathtt{Id}_A(a,b)\rho]_S\vec{\omega}) \mathbin{@} 1 = b\rho.$$

The uniformity conditions follow from those in A.

For a substitution $\sigma\colon \Delta \to \Gamma$ an element ω of $(\mathtt{Id}_A(a,b)\sigma)\rho = \mathtt{Id}_A(a,b)(\sigma\rho)$ is given by $\omega \mathbin{@} z$ in $A((\sigma\rho)s_z) = (A\sigma)(\rho s_z)$ with $\omega \mathbin{@} 0 = a(\sigma\rho) = (a\sigma)\rho$ and $\omega \mathbin{@} 1 = b(\sigma\rho) = (b\sigma)\rho$. Hence $\mathtt{Id}_A(a,b)\sigma = \mathtt{Id}_{A\sigma}(a\sigma, b\sigma)$ as types, and similarly this holds for the Kan structure. □

Note that for the filling operations in $\mathtt{Id}_A(a,b)$ we need the filling operations in A with *one more non-principal direction*. This is the main reason to require operations with non-principal sides!

Lemma 3.3.3. *For $\Gamma \vdash A$ and $\Gamma \vdash a : A$ we have $\Gamma \vdash \mathtt{refl}\, a : \mathtt{Id}_A(a,a)$, and $(\mathtt{refl}(a))\sigma = \mathtt{refl}(a\sigma)$ for a substitution $\sigma\colon \Delta \to \Gamma$.*

Proof. For $\rho \in \Gamma(I)$ define $(\mathtt{refl}\, a)\rho = a(\rho s_{x_I})$, i.e., $(\mathtt{refl}\, a)\rho = \langle x\rangle a(\rho s_x)$. This defines a term as $(a(\rho s_{x_I}))f = a(\rho s_{x_I}(f, x_I = x_J)) = a((\rho f)s_{x_J}) = (\mathtt{refl}\, a)(\rho f)$ for $f\colon I \to J$. □

Note that the previous lemma does not rely on $\Gamma \vdash A$ to be a Kan type.

Next, we will define an elimination operator for the identity type. We will define various operations which together define the J-eliminator where the usual definitional equality holds only up to propositional equality (i.e., we will give an inhabitant of the respective Id-type).

First we define the *transport* along a path. Let $\Gamma \vdash A$ be a type and $\Gamma.A \vdash C$ be a Kan type. Furthermore let $\Gamma \vdash a : A$, $\Gamma \vdash b : A$, $\Gamma \vdash e : C[a]$, and $\Gamma \vdash d : \mathtt{Id}_A(a,b)$. (Recall that $[a]$ is the substitution $(1,a)\colon \Gamma \to \Gamma.A$.) We define a term $\Gamma \vdash \mathtt{subst}_C(d,e) : C[b]$ as follows. For $\rho \in \Gamma(I)$ and a fresh $x = x_I$ we have that $d\rho \mathbin{@} x \in A\rho s_x$ with $(d\rho \mathbin{@} x)(x=0) = a\rho$ and $(d\rho \mathbin{@} x)(x=1) = b\rho$. Thus $(\rho s_x, d\rho \mathbin{@} x)$ connects $[a]\rho$ to $[b]\rho$ along x. We define

$$\mathtt{subst}_C(d,e)\rho = C(\rho s_x, d\rho \mathbin{@} x)^+_x(e\rho) \in C[b]\rho \tag{3.15}$$

where the composition operation is w.r.t. the shape $((x,1);\emptyset;I)$. This defines a term, since by the uniformity conditions we get for $f\colon I \to J$ and $y = x_J$ J-fresh:

$$\begin{aligned}(\mathtt{subst}_C(d,e)\rho)f &= \big(C(\rho s_x, d\rho \mathbin{@} x)^+_x(e\rho)\big)f \\ &= \big(C(\rho s_x, d\rho \mathbin{@} x)(f, x = y)\big)^+_y(e\rho f) \\ &= C(\rho f s_y, d\rho f \mathbin{@} y)^+_y(e\rho f) \\ &= \mathtt{subst}(d,e)(\rho f)\end{aligned}$$

where we used that $s_x(f, x = y) = f s_y$.

If $\sigma\colon \Delta \to \Gamma$, then $(\mathtt{subst}_C(d,e))\sigma = \mathtt{subst}_{C(\sigma \mathsf{p}, \mathsf{q})}(d\sigma, e\sigma)$ which is readily checked from the defining equation (3.15).

According to the definition of subst the line $C(\rho s_x, d\rho \mathbin{@} x)\!\uparrow_x(e\rho)$ connects $e\rho$ to $\mathtt{subst}_C(d,e)\rho$. In particular, for $d = \mathtt{refl}\, a$ we get

$$\Gamma \vdash \mathtt{substEq}_C(a,e) : \mathtt{Id}_{C[a]}(e, \mathtt{subst}_C(\mathtt{refl}\, a, e))$$

where

$$\mathtt{substEq}_C(a,e)\rho = \langle x\rangle\, C(\rho s_x, a\rho s_x)\!\uparrow_x(e\rho). \tag{3.16}$$

Similar to above one can show that this defines a term and is stable under substitution.

Definition 3.3.4. For a type $\Gamma \vdash A$ we define the type $\Gamma \vdash \mathtt{contr}(A)$ by

$$\mathtt{contr}(A) = \Sigma A\,(\Pi A \mathsf{p}(\mathtt{Id}_{A\mathsf{pp}}(\mathsf{qp}, \mathsf{q})))$$

or using "informal" type-theoretical notation

$$\mathtt{contr}(A) = (\Sigma x : A)\,(\Pi y : A)\ \mathtt{Id}_A(x, y).$$

We say that a type $\Gamma \vdash A$ is *contractible* if the type $\Gamma \vdash \mathtt{contr}(A)$ is inhabited. In this case we call the first projection a center of contraction.

Next, we show that if $\Gamma \vdash A$ is a Kan type and $\Gamma \vdash a : A$, then the *singleton type* $\mathtt{singl}(A, a) = \Sigma A\,\mathtt{Id}_{A\mathsf{p}}(a\mathsf{p}, \mathsf{q})$ is contractible. We clearly have $\Gamma \vdash (a, \mathtt{refl}\,a) : \mathtt{singl}(A, a)$. We now show that $(a, \mathtt{refl}\,a)$ is also a center of contraction, i.e., we have to give a term

$$\Gamma.\,\mathtt{singl}(A, a) \vdash \mathtt{isCenter}(a) : \mathtt{Id}((a, \mathtt{refl}\,a)\mathsf{p}, \mathsf{q}).$$

where we omitted the subscript $\mathtt{singl}(A, a)\mathsf{p}$. Let $\rho \in \Gamma(I)$ and $(b, \omega) \in \mathtt{singl}(A, a)\rho$, so $b \in A\rho$ and $\omega \in (\mathtt{Id}_{A\mathsf{p}}(a\mathsf{p}, \mathsf{q}))(\rho, b)$. For a fresh $x = x_I$, $\omega \mathbin{@} x \in A\rho s_x$ connects $a\rho$ to b. Let y be a fresh name ($y = x_{I,x}$). We have to give an element in $\mathtt{Id}((a, \mathtt{refl}\,a)\mathsf{p}, \mathsf{q})(\rho, (b, \omega))$ for which we give an element in $\mathtt{singl}(A, a)\rho s_y$ connecting $(a, \mathtt{refl}\,a)\rho$ to (b, ω). This amounts to give a pair whose first component α is in $A\rho s_y$ and connects a to b and whose second component in $(\mathtt{Id}_{A\mathsf{p}}(a\mathsf{p}, \mathsf{q}))(\rho s_y, \alpha)$ connects $(\mathtt{refl}\,a)\rho$ to ω along y. Consider the open box $(a\rho s_y, a\rho s_x, \omega \mathbin{@} x)$ in $A(\rho s_x s_y)$:

$$\begin{array}{ccc} & a\rho & \\ {\scriptstyle a\rho s_x}\uparrow & & \uparrow{\scriptstyle \omega \mathbin{@} x} \\ a\rho & \xrightarrow{a\rho s_y} & a\rho \end{array} \qquad b \tag{3.17}$$

Its filler gives rise to the second component and its composition, i.e., its upper face gives the first component. Thus we define

$$\begin{aligned} \mathtt{isCenter}(a)(\rho, (b, \omega)) = \langle y \rangle \big(& A(\rho s_x s_y)^{+}_{x,y}(a\rho s_y, a\rho s_x, \omega \mathbin{@} x), \\ & \langle x \rangle\, A(\rho s_x s_y)\!\uparrow_{x,y}(a\rho s_y, a\rho s_x, \omega \mathbin{@} x)\big) \end{aligned} \tag{3.18}$$

The uniformity conditions guarantee that this defines a section.

Let us define the more common elimination operator of Paulin-Mohring from the above operations—with the difference that the usual definitional equality is only propositional. To not make the notation too heavy we'll use informal reasoning in type theory; note that the definition can be given internally in type theory and we don't refer to the model; this definition follows Danielsson's Agda development[1] accompanying [29]. First note that using the transport operation `subst` one can define composition $p \mathbin{.} q : \mathtt{Id}_A(a, c)$ of two identity proofs $p : \mathtt{Id}_A(a, b)$, $q : \mathtt{Id}_A(b, c)$, as well as inverses $p^{-1} : \mathtt{Id}_A(b, a)$.

[1] Available at `www.cse.chalmers.se/~nad/`.

Let A be a type, $a : A$, and $C(b,p)$ a type given $b : A$, $p : \mathtt{Id}_A(a,b)$, such that $v : C(a, \mathtt{refl}\, a)$; for $b : A$ and $p : \mathtt{Id}_A(a,b)$ we define $\mathtt{J}(a,v,b,p) : C(b,u)$. We can consider C as a dependent type over $\mathtt{singl}(A,a)$ via $C(\mathtt{p}w, \mathtt{q}w)$ for $w : \mathtt{singl}(A,a)$. As we showed in the last paragraph, $\mathtt{singl}(A,a)$ is contractible with center $(a, \mathtt{refl}\, a)$, and thus we get a witness $\mathtt{app}(\varphi,(b,p)) : \mathtt{Id}((a,\mathtt{refl}\, a),(b,p))$ for $\varphi = \lambda\, \mathtt{isCenter}(a)$; now with $\mathtt{subst}$ (w.r.t. the type $C(\mathtt{p}w, \mathtt{q}w)$ for $w : \mathtt{singl}(A,a)$) we can define

$$\mathtt{J}(a,v,b,p) = \mathtt{subst}((\mathtt{app}(\varphi,(a,\mathtt{refl}\, a)))^{-1} \mathbin{\centerdot} \mathtt{app}(\varphi,(b,p)), v). \tag{3.19}$$

Note that we now are able to derive $\mathtt{Id}_{\mathtt{Id}_A(a,a)}(p^{-1} \centerdot p, \mathtt{refl}\, a)$ for all $p : \mathtt{Id}_A(a,b)$ using $\mathtt{J}$ and $\mathtt{substEq}$.

It remains to check the propositional equality for $\mathtt{J}$. If, in (3.19), $p = \mathtt{refl}\, a$ and $b = a$, we get that

$$\mathtt{app}(\varphi,(a,\mathtt{refl}\, a))^{-1} \mathbin{\centerdot} \mathtt{app}(\varphi,(b,p))$$

is propositionally equal to $\mathtt{refl}(\mathtt{refl}\, a)$, and thus using $\mathtt{subst}$ and $\mathtt{substEq}$ again one gets a witness of $\mathtt{Id}_{C(a,\mathtt{refl}\, a)}(v, \mathtt{J}(a,v,a,\mathtt{refl}\, a))$. This concludes the sketch that $\mathtt{J}$ with the rewrite rule as propositional equality is definable from $\mathtt{subst}, \mathtt{substEq}$, and $\mathtt{isCenter}$—and that alone in type theory without referring to their actual semantics.

Note that to get the propositional equality for $\mathtt{J}$ we could not use $\mathtt{subst}$ on $\mathtt{app}(\varphi,(b,p))$ directly.

3.3.1 Functional Extensionality

The equality on the function space is extensional.

Theorem 3.3.5. *Equality on* Π*-types is extensional, i.e., any pointwise equal functions are equal. More precisely, given* $\Gamma \vdash A$, $\Gamma.A \vdash B$ *we can justify the rule:*

$$\frac{\Gamma \vdash w' : \Pi AB \qquad \begin{array}{c}\Gamma \vdash w : \Pi AB \\ \Gamma \vdash e : \Pi A\ \mathtt{Id}_{B[q]}(\mathtt{app}(w\mathtt{p},\mathtt{q}), \mathtt{app}(w'\mathtt{p},\mathtt{q}))\end{array}}{\Gamma \vdash \mathtt{funExt}(w,w',e) : \mathtt{Id}(w,w')}$$

Proof. Let $\rho \in \Gamma(I)$, $x = x_I$ fresh, and write θ for $\mathtt{funExt}(w,w',e)$. We have to define a dependent function

$$(\theta\rho)_f \text{ in } \prod_{u \in A(\rho s_x f)} B(\rho s_x f, u) \text{ for each } f \colon I, x \to J.$$

In case $fx = 0$ we set $(\theta\rho)_f = w\rho_{(f-x)}$, and likewise, in case $fx = 1$ we set $(\theta\rho)_f = w'\rho_{(f-x)}$. For f defined on x we set $(\theta\rho)_f = (\theta(\rho f))_{\mathbf{1}}$ so that we can assume $f = \mathbf{1} \colon I, x \to I, x$. This definition ensures $(\theta\rho)g = \theta(\rho g)$. Let $u \in A\rho s_x$ and $u_b = u(x = b) \in A\rho$; we have to define $(\theta\rho)_{\mathbf{1}} u \in B(\rho s_x, u)$

connecting $w\rho_{\mathbf{1}}u_0$ to $w'\rho_{\mathbf{1}}u_1$ along x. Let y be fresh. We get that $(e\rho_{\mathbf{1}}u_1)\mathbin{@}y \in B(\rho s_y, u_1)$ and hence we define $(\theta\rho)_{\mathbf{1}}u$ to be the filler of the following box

$$\begin{array}{ccc} w\rho_{\mathbf{1}}u_0 & \xrightarrow{(\theta\rho)_{\mathbf{1}}u} & w'\rho_{\mathbf{1}}u_1 \\ {\scriptstyle (w\rho_{\mathbf{1}}u_0)s_y}\uparrow & & \uparrow{\scriptstyle (e\rho_{\mathbf{1}}u_1)\mathbin{@}y} \\ w\rho_{\mathbf{1}}u_0 & \xrightarrow[w\rho_{s_x}u]{} & w\rho_{\mathbf{1}}u_1 \end{array} \quad \text{over } B(\rho s_x s_y, us_y).$$

That is, $(\theta\rho)_{\mathbf{1}}u = B(\rho s_x s_y, us_y)^+_{y,x}(w\rho_{s_x}u, (w\rho_{\mathbf{1}}u_0)s_y, (e\rho_{\mathbf{1}}u_1)\mathbin{@}y)$. Now if $f\colon I, x \to J$ is defined on x and z J-fresh, we get by the uniformity conditions

$$\begin{aligned} ((\theta\rho)_{\mathbf{1}}u)f &= \big(B(\rho s_x s_y, us_y)^+_{y,x}(w\rho_{s_x}u, (w\rho_{\mathbf{1}}u_0)s_y, (e\rho_{\mathbf{1}}u_1)\mathbin{@}y)\big)f \\ &= \big(B(\rho s_x s_y, us_y)(f, y = z)\big)^+_{z,fx} \\ &\qquad \big((w\rho_{s_x}u)f, (w\rho_{\mathbf{1}}u_0)(f - x)s_z, (e\rho_{\mathbf{1}}u_1)(f - x)\mathbin{@}z\big) \end{aligned}$$

which is the same as $(\theta\rho)_f(uf)$. In case f is not defined on x we get $((\theta\rho)_{\mathbf{1}}u)f = (\theta\rho)_f(uf)$ by the above definition. We leave the verification of $(\theta\rho)f = \theta(\rho f)$ from these equations to the reader. □

3.3.2 Path Application

Although in general `subst` and `J` don't satisfy the usual definitional equalities the model justifies another operation `ap` which satisfies new definitional equalities which don't hold if we define the operation using `J`.

Theorem 3.3.6. *Let $\Gamma \vdash A$ and $\Gamma \vdash B$ be Kan types, $\Gamma \vdash u : A$, and $\Gamma \vdash v : A$. Then we can validate the rule*

$$\frac{\Gamma \vdash \varphi : A \to B \qquad \Gamma \vdash w : \mathtt{Id}_A(u, v)}{\Gamma \vdash \mathtt{ap}(\varphi, w) : \mathtt{Id}_B(\mathtt{app}(\varphi, u), \mathtt{app}(\varphi, v))}$$

satisfying

$$\begin{aligned} \mathtt{ap}(\mathrm{id}, w) &= w \\ \mathtt{ap}(\varphi \circ \psi, w) &= \mathtt{ap}\big(\varphi, \mathtt{ap}(\psi, w)\big) \\ \mathtt{ap}(\varphi, \mathtt{refl}\, a) &= \mathtt{refl}(\mathtt{app}(\varphi, a)) \\ \mathtt{ap}(\lambda(b\mathsf{p}), w) &= \mathtt{refl}\, b \end{aligned}$$

where id *is the identity function, $\circ$ denotes composition, and $\lambda(b\mathsf{p})$ is the constant b function. Moreover this operation is stable under substitution, i.e.,* $\mathtt{ap}(\varphi, w)\sigma = \mathtt{ap}(\varphi\sigma, w\sigma)$ *for $\sigma\colon \Delta \to \Gamma$.*

Proof. For $\rho \in \Gamma(I)$ and a fresh x we set

$$\mathtt{ap}(\varphi, w)\rho = \langle x \rangle\, (\varphi\rho)_{s_x}(w\rho \mathbin{@} x).$$

This defines a term as for $f\colon I \to J$ and y J-fresh we get

$$\begin{aligned}(\mathtt{ap}(\varphi, w)\rho)f &= \langle y\rangle\, ((\varphi\rho)_{s_x}(w\rho \mathbin{@} x))(f, x = y)\\ &= \langle y\rangle\, (\varphi\rho)_{s_x(f,x=y)}(w\rho \mathbin{@} x(f, x = y))\\ &= \langle y\rangle\, (\varphi\rho)_{fs_y}(w(\rho f) \mathbin{@} y)\\ &= \langle y\rangle\, (\varphi(\rho f))_{s_y}(w(\rho f) \mathbin{@} y) = \mathtt{ap}(\varphi, w)(\rho f)\end{aligned}$$

The other equations immediately follow from the definition. Let us, for example, check the second equation: $\varphi \circ \psi$ is $\lambda\, \mathtt{app}(\varphi\mathtt{p}, \mathtt{app}(\psi\mathtt{p}, \mathtt{q}))$ and hence

$$\begin{aligned}\mathtt{ap}(\varphi \circ \psi, w)\rho \mathbin{@} x &= ((\varphi \circ \psi)\rho)_{s_x}(w\rho \mathbin{@} x)\\ &= \mathtt{app}(\varphi\mathtt{p}, \mathtt{app}(\psi\mathtt{p}, \mathtt{q}))(\rho s_x, w\rho \mathbin{@} x)\end{aligned}$$

which using $((\varphi\mathtt{p})(\rho s_x, w\rho \mathbin{@} x))_{\mathbf{1}} = (\varphi\rho)_{s_x}$ (and analogously for ψ) becomes

$$\begin{aligned}&= (\varphi\rho)_{s_x}((\psi\rho)_{s_x}(w\rho \mathbin{@} x))\\ &= (\varphi\rho)_{s_x}(\mathtt{ap}(\psi, w)\rho \mathbin{@} x)\\ &= (\mathtt{ap}(\varphi, \mathtt{ap}(\psi, w)))\rho \mathbin{@} x.\end{aligned}$$

□

3.3.3 Heterogeneous Identity Types

The model also comes with a natural notion of *(weak) heterogeneous identity types* satisfying the following rules

$$\frac{\begin{array}{c}\Gamma \vdash A \qquad \Gamma \vdash u_0 : A \qquad \Gamma \vdash u_1 : A \qquad \Gamma \vdash p : \mathtt{Id}_A(u_0, u_1)\\ \Gamma.A \vdash C \qquad \Gamma \vdash v_0 : C[u_0] \qquad \Gamma \vdash v_1 : C[u_1]\end{array}}{\Gamma \vdash \mathtt{HId}^p_C(v_0, v_1)}$$

$$\mathtt{HId}^{\mathtt{refl}\, u}_C(v_0, v_1) = \mathtt{Id}_{C[u]}(v_0, v_1)$$

omitting the equations for stability under substitution. Its interpretation in the cubical set model is given by: for $\rho \in \Gamma(I)$ the set $(\mathtt{HId}^p_C(v_0, v_1))\rho$ contains elements $\langle x\rangle w$ with $w \in C(\rho s_x, p\rho \mathbin{@} x)$ (up to renaming of bound variables, or making a canonical choice $x = x_I$ as for the identity type) such that $w(x = b) = v_b$ for $b \in \mathbf{2}$. The Kan structure is given by the equation:

$$[(\mathtt{HId}^p_C(v_0, v_1))\rho]_S\vec{\omega} = \langle z\rangle\, [C(\rho s_z, p\rho \mathbin{@} z)]_{S,z}(\vec{\omega}, v_0\rho, v_1\rho).$$

The accompanying equation is immediate given the definition of $\mathtt{Id}$.

3.4 Regular Kan Types

We have seen that in general the usual definitional equality for $\mathtt{J}$ holds only up to propositional equality. If we restrict to Kan types which satisfy a regularity condition we can give a definition of $\mathtt{J}$ for which the equality is definitional. This notion is however *not* preserved by all the type formers.

Definition 3.4.1. A Kan type $\Gamma \vdash A$ is *regular* if for any open box shape S with principal direction x and S-open box $\vec{u}$ in ρs_x such that each component u_{yb} with $y \neq x$ is degenerated along x, i.e., $u_{yb} = v_{yb} s_x$ for some $v_{yb} \in A\rho(y = b)$, then the filling satisfies

$$[A\rho s_x]_S \vec{u} = u_{xa} s_x$$

where u_{xa} is the principal face of $\vec{u}$. (So $\vec{u} = u_{xa}, \vec{v} s_x$.)

Theorem 3.4.2. *Regularity is preserved under substitution and the type formers* `Id` *and* Σ*, that is:*

1. *If* $\Gamma \vdash A$ *is a regular Kan type, then so is* $\Delta \vdash A\sigma$ *for* $\sigma \colon \Delta \to \Gamma$.
2. *If* $\Gamma \vdash A$ *be a regular Kan type,* $\Gamma \vdash a : A$*, and* $\Gamma \vdash b : A$*, then also* $\Gamma \vdash \mathtt{Id}_A(a, b)$ *is regular.*
3. *If* $\Gamma \vdash A$ *and* $\Gamma.A \vdash B$ *are regular Kan types, then so is* $\Gamma \vdash \Sigma AB$.

Proof. The proof is by analyzing the definitions of the fillings. For (1) recall that the fillings in $(A\sigma)\rho s_x$ are defined by fillings in $A(\sigma(\rho s_x))$. Since σ commutes with degenerates, regularity is preserved to $A\sigma$.

For (2) the defining equation (3.14) yields for an S-open box $\omega, \vec{\omega}$ in $\mathtt{Id}_A(a, b)\rho s_x$ degenerate along x (where x is the principal direction of S and ω the principal side of the box)

$$\begin{aligned}[\mathtt{Id}_A(a, b)\rho s_x]_S(\omega, \vec{\omega}) &= \langle z \rangle\, [A\rho s_x s_z]_{S,z}(\omega \mathbin{@} z, \vec{\omega} \mathbin{@} z, a\rho s_x, b\rho s_x) \\ &= \langle z \rangle\, (\omega \mathbin{@} z) s_x = \omega s_x\end{aligned}$$

as $s_x s_z = s_z s_x$ and $(\omega \mathbin{@} z, \vec{\omega} \mathbin{@} z, a\rho s_x, b\rho s_x)$ is also degenerate along x.

For (3) one readily checks from the defining equations in Theorem 3.2.3. □

Let us sketch how one can use regularity in order to define a variant of `J` given in Section 3.3 but with the right definitional equality for regular Kan types (using the notations from Section 3.3): First, the definition of `subst` and `isCenter` is as in Section 3.3. Note that by the regularity condition (for C) $\mathtt{substEq}_C(a, e)$ is simply $\mathtt{refl}(e)$ in equation (3.16) on page 56. Moreover, the definition of $\mathtt{isCenter}(a)$ on page 57 is such that if $b = a\rho$ and $\omega = a\rho s_x$ the square (3.17) is degenerate. Second, this can be used to directly define a variant $\mathtt{J}'$ of `J` by

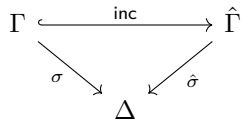
$$\mathtt{J}'(a, v, b, p) = \mathtt{subst}(\mathtt{app}(\varphi, (b, p)), v).$$

where φ was $\lambda\, \mathtt{isCenter}(a)$. This now satisfies the right definitional equality since if $p = \mathtt{refl}(a)$ and $b = a$, then $\mathtt{app}(\varphi, (a, \mathtt{refl}(a))) = \mathtt{refl}(a, \mathtt{refl}(a))$, and so using `subst` along a reflexivity path yields v.

3.5 Kan Completion

In this section we will show how to complete any cubical set Γ to a Kan cubical set $\hat{\Gamma}$. This works by freely attaching fillers to Γ; their restrictions are guided by the uniformity conditions. This construction however does not work for dependent types in a satisfactory way since it does not necessarily commute with substitutions.[2]

Theorem 3.5.1. *For any cubical set Γ there is a Kan cubical set $\hat{\Gamma}$ and a monomorphism $\mathsf{inc}\colon \Gamma \to \hat{\Gamma}$ such that for any Kan cubical set Δ and $\sigma\colon \Gamma \to \Delta$ there is a morphism $\hat{\sigma}\colon \hat{\Gamma} \to \Delta$ making the following diagram commute:*



Proof. Given a cubical set $\Gamma \vdash$ we define the sets $\hat{\Gamma}(I)$ for I in $\square$ and restriction maps $\hat{\Gamma}(I) \ni \rho \mapsto \rho f \in \hat{\Gamma}(J)$ for $f\colon I \to J$ by induction-recursion as follows. The sets are given by the rules:

1. If $\rho \in \Gamma(I)$, then $\mathsf{inc}\,\rho \in \hat{\Gamma}(I)$.

2. If S is an open box shape with principal side (x, a) and $\vec{u}$ is an S-open box in $\hat{\Gamma}(I)$, then $\mathsf{fill}_S\,\vec{u} \in \hat{\rho}$.

3. If S is an open box shape with principal side (x, a) and $\vec{u}$ is an S-open box in $\hat{\Gamma}(I)$ *and* $x = x_{I-x}$, then $\mathsf{comp}_S\,\vec{u} \in \hat{\Gamma}(I - x)$.

Here inc, fill_S, and comp_S are *constructors*, with the intended rôle for the latter two being the filling and composition operation, respectively. Note that in (2) and (3) being an open box refers to the restrictions defined at the same time. Note that the assumption on the variable x in (3) is due to the fact that x is bound. (One could also identify these expressions up to renaming of the bound variables, similar as for the path types.) The restrictions are guided by the uniformity conditions. For $f\colon I \to J$ we define

$$(\mathsf{inc}\,a)f = \mathsf{inc}(af)$$

$$(\mathsf{fill}_S\,\vec{u})f = \begin{cases} u_{yc}(f - y) & \text{if for some } (y, c) \in \langle S\rangle, fy = c, \\ \mathsf{comp}_{Sf'}(\vec{u}f') & \text{if } fx = a, \\ \mathsf{fill}_{Sf}(\vec{u}f) & \text{otherwise.} \end{cases}$$

[2] Indeed, type theory can't be consistently extended with such a rule, cf. `http://ncatlab.org/homotopytypetheory/show/Homotopy+Type+System#fibrant_replacement` (October 13, 2014).

where $f' = (f - x, x = x_J)$ and the restriction of a composition $\mathsf{comp}_S\,\vec{u}$ along $f : I - x \to J$ is defined by

$$(\mathsf{comp}_S\,\vec{u})f = \begin{cases} u_{yc}(x = a)(f - y) & \text{if for some } (y, c) \in \langle S\rangle,\ fy = c, \\ \mathsf{comp}_{S\tilde{f}}(\vec{u}\tilde{f}) & \text{otherwise.} \end{cases}$$

where now $\tilde{f} = (f, x = x_J) : I \to J, x_J$.

Now one proves by induction on $a \in \hat{\Gamma}(I)$ that $(\rho f)g = \rho(fg)$ and $\rho\mathbf{1} = \rho$ to make $\hat{\Gamma}$ into a cubical set. The Kan fillers are given by the constructor fill_S making $\hat{\Gamma}$ into a Kan cubical set. We directly get the monomorphism $\Gamma \to \hat{\Gamma}$ from the constructor inc.

Given a Kan cubical set Δ and $\sigma : \Gamma \to \Delta$ we define maps $\hat{\Gamma}(I) \to \Delta(I), \rho \mapsto \hat{\sigma}\rho$ while simultaneously proving $(\hat{\sigma}\rho)f = \hat{\sigma}(\rho f)$ for $f : I \to J$:

$$\begin{aligned} \hat{\sigma}(\mathsf{inc}\,\rho) &= \sigma\rho \\ \hat{\sigma}(\mathsf{fill}_S\,\vec{u}) &= [\Delta]_S(\hat{\sigma}\vec{u}) \\ \hat{\sigma}(\mathsf{comp}_S\,\vec{u}) &= |\Delta|_S(\hat{\sigma}\vec{u}) \end{aligned}$$

where (x, a) is the principle side of S (with $x = x_{I-x}$ in the last case), and $\hat{\sigma}\vec{u}$ is the family of elements $\hat{\sigma}(u_{yb})$ for $(y, b) \in \langle S\rangle$; by the induction hypothesis, this is an open box in $\Delta(I)$. That $\hat{\varphi}$ commutes with restrictions is by construction, as is the commuting diagram. □

Chapter 4

The Universe of Kan Cubical Sets

In this chapter we will define the universe of small Kan types and show that it is itself a uniform Kan cubical set. The main work goes into the latter and we will decompose this into first defining the composition- and then the filling operations (similarly to what we did for Π-types).

Recall from Section 1.2.4 how to lift a Grothendieck universe $\mathbf{Set}_0$ to a universe in the presheaf model. This is adapted to give a universe of small Kan types by basically replacing "type" with "Kan type" in the definition: First, we adapt the definition of small type. The judgment $\Gamma \vdash A\ \mathsf{KType}_0$ is defined to be that $\Gamma \vdash A$ is a Kan type and $A\rho$ is a small set (i.e., an element of $\mathbf{Set}_0$) for each $\rho \in \Gamma(I)$; we also call such a $\Gamma \vdash A$ a *small Kan type*. Second, we define the universe accordingly:

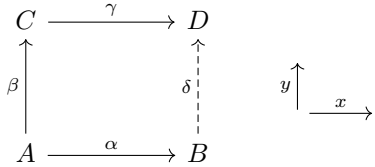
Definition. The cubical set $\mathbf{U}$ of small Kan types is defined as follows. The set $\mathbf{U}(I)$ consists of all small Kan types $\mathbf{y}I \vdash A\ \mathsf{KType}_0$; restrictions along $f : I \to J$ are defined by substituting with $\mathbf{y}f : \mathbf{y}J \to \mathbf{y}I$.

The definitions of $\ulcorner\cdot\urcorner$ and El are as in Section 1.2.4, where the fillings are defined by $[(\mathrm{El}\, T)\rho]_S \vec{u} = [(T\rho)_{\mathbf{1}}]_S \vec{u}$ and $[(\ulcorner A \urcorner \rho)_f]_S \vec{u} = [A(\rho f)]_S \vec{u}$. This defines a universe structure on the Kan cubical set model if we prove that $\Gamma \vdash \mathbf{U}$ is itself a Kan type, i.e., that $\mathbf{U}$ is a Kan cubical set.

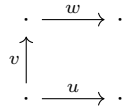
Note that the points of $\mathbf{U}$ are simply the (small) Kan cubical sets: for $\mathbf{y}\emptyset \vdash A$ we get a Kan cubical set with $A(I)$ being A_f where f is the unique $\emptyset \to I$. A line in $\mathbf{U}$ between points A and B can be seen as a "heterogeneous" notion of lines, cubes, ..., $a \to b$ where a is an I-cube of A and b and I-cube of B.

The goal of the rest of this chapter is to prove that the cubical set $\mathbf{U}$ of small Kan types is a Kan cubical set. We first show that $\mathbf{U}$ has compositions. The intuitive idea behind the composition is that of composing relations (hence

the name). If we are given a open box in the universe, say of the form



we want to define the line δ in the universe. δ will be given by a family δ_f for $f\colon \{y\} \to I$ where the main case is $\delta_{\mathbf{1}}$ which will be defined to consist of triples (u, v, w) where $u \in \alpha_{\mathbf{1}}$, $v \in \beta_{\mathbf{1}}$, and $w \in \gamma_{\mathbf{1}}$ such that they are compatible in the sense that $u(x = 0) = v(y = 0)$ and $w(x = 0) = v(y = 1)$, i.e., a open box shape:



We then have to verify that δ has filling operations.

Lemma. **U** *has composition operations.*

Proof. Let $S = ((x, d); J; I)$ be an open box shape in I and $\vec{A}$ an S-open box in $\mathbf{U}(I)$, that is, compatible $A_{ya} \in \mathbf{U}(I - y)$ for $(y, a) \in \langle S\rangle$. We first define the composition $A_{xd} = |\mathbf{U}|_S \vec{A}$ as a type $\mathbf{y}(I - x) \vdash A_{xd}$ and then explain its Kan structure. Let $\bar{d} = 1 - d$.

Before we define A_{xd} let us introduce some notation. An S-open box $\vec{u}$ in $\vec{A}$ is given by a family $u_{yb} \in (A_{yb})_{\mathbf{1}}$ for $(y, b) \in \langle S\rangle$ such that they are compatible, i.e., $u_{yb}(z = c) = u_{zc}(y = b) \in (A_{yb})_{(z=c)} = (A_{zc})_{(y=b)}$ for $y \neq z$, $(y, b), (z, c) \in \langle S\rangle$. We denote the set of all such S-open boxes by $\langle S\rangle\vec{A}$. Note that if $f\colon I \to K$ is defined on x, J, then $\vec{u} \in \langle S\rangle\vec{A}$ implies $\vec{u}f \in \langle Sf\rangle\vec{A}f$.

For $f\colon I - x \to K$ we define the (small) set $(A_{xd})_f$ by distinguishing cases. In case $f(y) = b$ for some $(y, b) \in \langle S\rangle$ (note that $y \neq x$), then $f = (y = b)(f - y)$ and define $(A_{xd})_f = (A_{yb})_{(f-y,x=d)}$. Note that this is well defined as $\vec{A}$ is compatible. Otherwise, i.e., in case f is defined on J, we define $(A_{xd})_f = \langle Sf'\rangle Af'$ where $f' = (f, x = x_K)\colon I \to K, x_K$. This guarantees that A_{xd} has the right faces. One can also define it to be elements $\langle z\rangle\vec{u}$ with $\vec{u} \in \langle S(f, x = z)\rangle\vec{A}(f, x = z)$ and identify modulo α-conversion; we will use this notation.

To summarize, $(A_{xd})_f$ consists of elements of the form

1. $u \in (A_{yb})_{(f-y,x=d)}$ if $f(y) = b$ for some $(y, b) \in \langle S\rangle$;

2. $\langle z\rangle\vec{u}$, where $\vec{u} \in \langle S(f, x = z)\rangle\vec{A}(f, x = z)$ and z is fresh, otherwise.

Now if $g\colon K \to L$ we define the restrictions of an element in $(A_{xd})_f$ as follows. For elements u of the form (1), we use the restriction ug of $(A_{yb})_{(f-y,x=d)}$.

The restriction of an element $\langle z\rangle\vec{u}$ of the form (2) is defined by

$$(\langle z\rangle\vec{u})g = \begin{cases} u_{(fy)b}(g - fy, z = d) & \text{if } g(f(y)) = b \text{ for some } (y,b) \in \langle S\rangle, \\ \langle z'\rangle\vec{u}(g, z = z') & \text{otherwise,} \end{cases}$$

where z' is fresh w.r.t. the codomain of g. Note that in each case the resulting element is in $(A_{xd})_{fg}$. In particular, for $(y,b) \in \langle S\rangle$ and $f = \mathbf{1}$ we have

$$(\langle z\rangle\vec{u})(y = b) = u_{yb}(z = d).$$

This defines $A_{xd} = |\mathbf{U}|_S\vec{A}$ as a cubical set satisfying (as cubical sets)

$$(|\mathbf{U}|_S\vec{A})(y = b) = A_{yb} \qquad \text{for } (y,b) \in \langle S\rangle, \tag{4.1}$$

$$(|\mathbf{U}|_S\vec{A})f = |\mathbf{U}|_{Sf'}\vec{A}f' \qquad \text{if } f \text{ is defined on } J, \tag{4.2}$$

with $f' = (f, x = z)$ and in particular as sets we have

$$(|\mathbf{U}|_S\vec{A})_f = \begin{cases} (A_{yb})_{(f-y,x=d)} & \text{if } f(y) = b \text{ for some } (y,b) \in \langle S\rangle, \\ (|\mathbf{U}|_{Sf'}\vec{A}f')_{\mathbf{1}} & \text{if } f \text{ is defined on } J. \end{cases} \tag{4.3}$$

We now have to define the filling operations for $(A_{xd})_f = (|\mathbf{U}|_S\vec{A})_f$. W.l.o.g. we assume that $f = \mathbf{1}\colon I - x \to I - x$ as we take (4.3) as defining equations for the filling operations as well otherwise. Let $\vec{w}$ be an open box of shape $S' = ((x',d'); J'; I - x)$ in $(A_{xd})_{\mathbf{1}}$, i.e., $w_{yb} \in (A_{xd})_{(y=b)}$ for $(y,b) \in \langle S'\rangle$ such that for $(y,b),(z,c) \in \langle S'\rangle$, $y \neq z$

$$w_{yb}(z = c) = w_{zc}(y = b). \tag{4.4}$$

Note that for $(y,b) \in \langle S'\rangle - \langle S\rangle$ (i.e., $(y,b) \in \langle S'\rangle$ and $y \notin J$) we have that $w_{yb} = \langle x\rangle\vec{u}^{yb}$ with $\vec{u}^{yb} \in \langle S(y = b)\rangle\vec{A}(y = b)$. (We assume that all bound variables are x which is fresh for $I - x$.) Since $y \notin x, J$ we have $\langle S(y = b)\rangle = \langle S\rangle$, so $u^{yb}_{zc} \in (A_{zc})_{(y=b)}$ for $(z,c) \in \langle S\rangle$ such that

$$u^{yb}_{zc}(z' = c') = u^{yb}_{z'c'}(z = c) \tag{4.5}$$

whenever these elements are defined. Moreover, by the definition of the restriction

$$w_{yb}(z = c) = u^{yb}_{zc}(x = d) \tag{4.6}$$

and so, since $\vec{w}$ is adjacent compatible (4.4), we get for $(y,b) \in \langle S'\rangle - \langle S\rangle$ and $(z,c) \in \langle S\rangle$,

$$w_{zc}(y = b) = u^{yb}_{zc}(x = d). \tag{4.7}$$

Moreover, if $(y,b),(y',b') \in \langle S'\rangle - \langle S\rangle$ with $y \neq y'$, we have since $w_{yb}(y' = b') = w_{y'b'}(y = b)$ that the corresponding entries at $(z,c) \in \langle S\rangle$ of the vectors are equal, i.e.,

$$u^{yb}_{zc}(y' = b') = u^{y'b'}_{zc}(y = b). \tag{4.8}$$

For $(z,c) \in \langle S\rangle$ we denote the family (not necessarily an open box) of the u^{yb}_{zc}, $(y,b) \in \langle S'\rangle - \langle S\rangle$ by $\mathrm{aux}(z,c)$. By (4.8) this family is compatible.

We want to define the element $[(A_{xd})_{\mathbf{1}}]_{S'}\vec{w} = [(|\mathbf{U}|_S\vec{A})_{\mathbf{1}}]_{S'}\vec{w} \in (A_{xd})_{\mathbf{1}}$ in such a way that this definition satisfies the uniformity condition. I.e., for $f\colon I - x \to L$ defined on $x', J' \subseteq I - x$, we require

$$([(|\mathbf{U}|_S\vec{A})_{\mathbf{1}}]_{S'}\vec{w})f = [(|\mathbf{U}|_S\vec{A})_f]_{S'f}(\vec{w}f)$$

that is, according to (4.3),

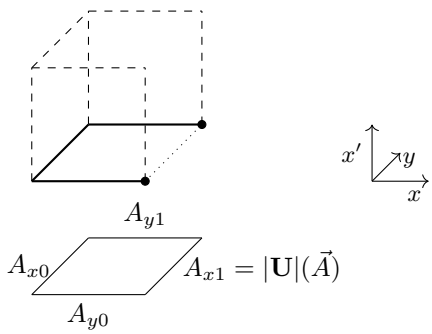
$$\big([(|\mathbf{U}|_S\vec{A})_{\mathbf{1}}]_{S'}\vec{w}\big)f = \begin{cases} [(|\mathbf{U}|_{Sf'}\vec{A}f')_{\mathbf{1}}]_{S'f}(\vec{w}f) & \text{if } J \subseteq \mathrm{def}(f), \\ [(A_{yb})_{(f-y,x=d)}]_{S'f}(\vec{w}f) & \text{if } fy = b \text{ for a } (y,b) \in \langle S\rangle. \end{cases} \tag{4.9}$$

Let us write $\langle x\rangle\vec{u}$ for the element $[(|\mathbf{U}|_S\vec{A})_{\mathbf{1}}]_{S'}\vec{w}$ we are going to define. To satisfy the second equation of (4.9) we need for $y \in J$ and $(y,b) \notin \langle S\rangle'$ that

$$u_{yb}(x = d) = (\langle x\rangle\vec{u})(y = b) = [(A_{yb})_{(x=d)}]_{S'(y=b)}(\vec{w}(y = b)). \tag{4.10}$$

To give $\langle x\rangle\vec{u}$ there are three cases to consider. We leave it for the reader to verify $(\langle x\rangle\vec{u})(y = b) = w_{yb}$ for $(y,b) \in \langle S\rangle$ along with the definition of $\vec{u}$.

1. *W.l.o.g.* $J \subseteq x', J'$. Let us first illustrate this in a (low-dimensional) special case where $I = \{x, x', y\}$, $J = \{y\}$, $d = d' = 1$, and also $J' = \emptyset$. We are given the dotted line in:



The types of the corresponding cubes are indicated in the lower square (which is not filled). The dotted line is, as an element in the composition, given by an open box indicated as the solid lines. To give the filling of the dotted line in the upwards direction is to give an open box indicated with the dashed lines; the first step is to fill each of the black dots upwards, and to proceed with the other cases with the extended box which now contains the non-principal sides for $y \in J$.

More formally and in the general case, for each $y \in J$ with $y \notin x', J'$ we have that $(y,b) \notin \langle S\rangle'$ (for $b \in \mathbf{2}$) and we construct

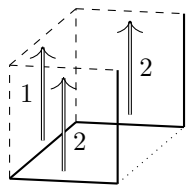
$$w_{yb} \in (A_{yb})_{(x=d)}$$

by filling $\vec{w}(y=b)$ of shape $S'(y=b)$ in $(A_{yb})_{(x=d)}$. Note that for $(y,b),(z,c)$ with $y,z \in J$ and $y,z \notin x',J'$ the so constructed elements are adjacent compatible since:

$$\begin{aligned} w_{yb}(z=c) &= ([(A_{yb})_{(x=d)}]_{S'(y=b)}(\vec{w}(y=b)))(z=c) \\ &= [(A_{yb})_{(x=d)(z=c)}]_{S'(y=b)(z=c)}(\vec{w}(y=b)(z=c)) \\ &= [(A_{zc})_{(x=d)(y=b)}]_{S'(z=c)(y=b)}(\vec{w}(z=c)(y=b)) \\ &= w_{zc}(y=b) \end{aligned}$$

Moreover, by construction they are adjacent compatible with the given open box $\vec{w}$. Thus, we can extend the $\vec{w}$ to a $((x,d);J',(J-(x',J'));I-x)$ open box.

2. *Case* $x' \notin J$. Let us first illustrate again in the special case as above but now with $J'=y$. So we are given the dotted line and the the solid lines on the right in:



The dotted line again corresponds to the three lower solid lines, and we want to construct three squares indicated by the dashed lines. To do so, we first fill on the left as indicated by the double arrow labeled "1"; second, we fill those sides labeled with "2" by the other sides taking into account those faces already constructed in the first step.

2.1. We construct $u_{x\bar{d}} \in (A_{x\bar{d}})_{\mathbf{1}}$ by filling $\mathrm{aux}(x,\bar{d})$ in $(A_{x\bar{d}})_{\mathbf{1}}$. Note that here $\mathrm{aux}(x,\bar{d})$ is an open box of shape $((x',d');J'-J;I-x)$.

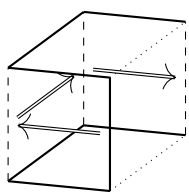
2.2. Next, for $(z,c) \in \langle S \rangle$ with $z \neq x$ we construct $u_{zc} \in (A_{x\bar{d}})_{\mathbf{1}}$ by filling in $(A_{x\bar{d}})_{\mathbf{1}}$ the open box

$$\mathrm{aux}(z,c), u_{x\bar{d}}(z=c), w_{zc} \text{ of shape } ((x',d');(J'-J),x;I-z).$$

where the latter two elements are the non-principal sides at $(x,\bar{d})$ and (x,d), respectively.

2.3. This concludes the construction of $\vec{u}$ in this case.

3. *Case* $x' \in J$. Let us again first sketch the construction in the above special case where $x'=y$, $J=J'=y$. We are given the right hand open box given by the two dotted lines (which become the lower and upper solid lines) and the solid line on the right in:



The filler of the open box is now constructed doing the fillers indicated with the double arrows in order: starting with the front where there are three solid lines forming an open box, and continuing the way to the back, always taking into account the face of the previously constructed filler as principal side.

3.1. We begin by extending the input box along x in direction $\bar{d}$. More precisely, we construct $u_{zc} \in (A_{zc})_{\mathbf{1}}$ where $(z,c) \in \langle (x',d'); J - x' \rangle$ (note $J - x' = J \cap J'$) by filling

$$w_{zc}, \mathrm{aux}(z,c) \text{ of shape } ((x,\bar{d}); J' - J; I - z).$$

Here $w_{zc} \in (A_{xd})_{zc} = (A_{zc})_{(x=d)}$ is the principal side of the open box; moreover, note that $\langle (x',d'); J - x' \rangle$ contains $(x',\bar{d'})$ where $\bar{d'} = 1 - d'$, but not (x',d') and not $(x,\bar{d})$.

3.2. Next, we construct $u_{x\bar{d}} \in (A_{x\bar{d}})_{\mathbf{1}}$ by filling the open box

$$\begin{array}{l} u_{zc}(x = \bar{d}) \text{ for } (z,c) \in \langle (x',d'); J - x' \rangle, \\ \mathrm{aux}(x,\bar{d}) \end{array}$$

of shape

$$((x',d'); (J - x') \cup (J' - J); I - x) = ((x',d'); J'; I - x) = S',$$

where $u_{x'\bar{d'}}(x = \bar{d}) \in (A_{x'\bar{d'}})_{(x=\bar{d})} = (A_{x\bar{d}})_{(x'=\bar{d'})}$ is the principal side of the open box.

3.3. Finally, we construct the missing side $u_{x'd'} \in (A_{x'd'})_{\mathbf{1}}$ by filling

$$\begin{array}{l} u_{x\bar{d}}(x' = d'), \\ u_{zc}(x' = d') \text{ for } z \in J - x',\ c \in \mathbf{2}, \text{ and} \\ \mathrm{aux}(x',d') \end{array}$$

of shape $((x,d); (J - x) \cup (J' - J); I - x') = ((x,d); J'; I - x')$.

3.4. This concludes the construction of $\vec{u}$ in this case.

We have to verify that this definition satisfies the uniformity conditions, i.e., that equations (4.9) are valid. Let $f \colon I - x \to L$ be defined on x', J'.

Assume $fy = b$ for some $y \in J$. To simplify notations, say $f = (z = b)$. Then $y \notin x', J'$ since f was defined on x', J'. Thus we obtain

$$(\langle x \rangle \vec{u})(y = b) = w_{yb} = [(A_{yb})_{(x=d)}]_{S'(x=d)}(\vec{w}(x = d))$$

as constructed in step 1, which we had to show.

Let us now assume that f is also defined on J. We have to show

$$(\langle x \rangle \vec{u})f = [(|\mathbf{U}|_{Sf'} \vec{A} f')_{\mathbf{1}}]_{S'f}(\vec{w} f)$$

where $f' = (f, x = x^*)$ with x^* fresh. Let us denote the right hand side element by $\langle x^* \rangle \vec{u}^*$ and all abbreviations used in the definition of $\vec{u}^*$ will be decorated with a $*$ as well (e.g., aux^*). Thus we have to show $\vec{u} f' = \vec{u}^*$. Since f is injective $x' \in J$ iff $fx' \in fJ$, and thus $\vec{u}$ and $\vec{u}^*$ are defined via the same

case. Moreover, for $(y,b) \in \langle S' \rangle - \langle S \rangle$ (which is iff $(fy,b) \in \langle S'f \rangle - \langle Sf' \rangle$) we have

$$w^*_{(fy)b} = w_{yb}(f-y) = (\langle x \rangle \vec{u}^{yb})(f-y) = \langle x^* \rangle\, \vec{u}^{yb}(f-y, x=x^*)$$

and thus $\mathrm{aux}(z,c)(f'-z) = \mathrm{aux}^*(f'z,c)$ for $(z,c) \in \langle S \rangle$. Now for example, in case 2 the first construction of $u^*_{(fx)\bar{d}}$ is by filling $\mathrm{aux}^*(fx,\bar{d})$, so

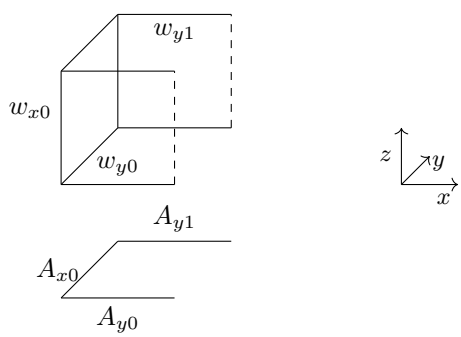
$$\begin{aligned} u^*_{x^*\bar{d}} &= [(A^*_{x^*\bar{d}})_{\mathbf{1}}](\mathrm{aux}^*(x^*,\bar{d})) \\ &= [(A^*_{x^*\bar{d}})_{\mathbf{1}}](\mathrm{aux}(x,\bar{d})(f'-x)) \\ &= [(A_{x\bar{d}})_f](\mathrm{aux}(x,\bar{d})f) \\ &= u_{x\bar{d}}f = u_{x\bar{d}}(f'-x) \end{aligned}$$

using the uniformity condition of $(A_{x\bar{d}})_{\mathbf{1}}$. Similarly, in the construction of the $u^*_{(fy)b}$ for $(y,b) \in \langle S \rangle$, $y \neq x$. The other case is analogous, concluding the proof. □

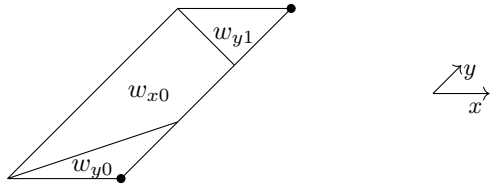
Theorem. **U** *is a Kan cubical set.*

Proof. We extend the composition operations of the previous lemma to filling operations making **U** into a Kan cubical set. Let $S = ((x,d);J;I)$ be an open box shape in I and $\vec{A}$ an S-open box in $\mathbf{U}(I)$, i.e., adjacent compatible $A_{ya} \in \mathbf{U}(I-y)$ for $(y,a) \in \langle S \rangle$. We first define the filling $A = [\mathbf{U}]_S\vec{A}$ as a type $\mathbf{y}I \vdash A_{xd}$ and then explain its Kan structure. Note that this will be such that $A(x=d) = A_{xd}$ with $A_{xd} := |\mathbf{U}|_S\vec{A}$ as constructed in the preceding lemma.

For $f\colon I \to K$ we give a (small) set A_f. In case $fy = b$ for some $(y,b) \in \langle S \rangle \cup \{(x,d)\}$ we have $f = (y=b)(f-y)$ and set $A_f = (A_{yb})_{(f-y)}$. Otherwise, i.e., f is defined on x, J, we can w.l.o.g. assume $f = \mathbf{1}\colon I \to I$ as we otherwise set $A_f = ([\mathbf{U}]_{Sf}\vec{A}f)_{\mathbf{1}}$. The set $A_{\mathbf{1}}$ is now defined as follows: an element is of the form $\langle z \rangle \vec{w}$ where $\vec{w}$ is an open box in $\vec{A}s_z$ of shape Ss_z and z is fresh, so $\vec{w}$ is given by elements $w_{yb} \in (A_{yb})_{s_z}$ for $(y,b) \in \langle S \rangle$; moreover, we require for $(y,b) \in \langle S \rangle$ with $y \neq x$ that $w_{yb}(x=1) \# z$, i.e., w_{yb} is degenerate along z. Here z is a bound variable. Let us illustrate this definition in the special case where $J = y$ and $d = 1$: such an element $\vec{w}$ is given by



where the dashed lines are required to be degenerate. Note that projecting $\vec{w}$ to $(z = 1)$ gives an element of $A_{x1} = |\mathbf{U}|(\vec{A})$ (disregarding binders for the moment). A good way to think of $\vec{w}$ is to imagine w_{y0} and w_{y1} as triangles by shrinking each of the dashed lines to a point, and to think of the dimension z as "hidden":



Here, the right hand lines are the projection $(z = 1)$, which is how we will define the restriction of the element $\vec{w}$ to $(x = 1)$, and the dots correspond to the dashed lines above.

The restriction $ug \in A_{fg}$ along $g\colon K \to L$ of an element $u \in A_f$ where $f\colon I \to K$ is defined as follows: In case $fy = b$ for some $(y, b) \in \langle S\rangle \cup \{(x, d)\}$, ug is given by the restriction of $(A_{yb})_{(f-y)}$. Otherwise, f is defined on x, J and $u = \langle z\rangle\vec{w}$ with $\vec{w}$ and open box in $\vec{A}f$ of shape $(Sf)s_z$; we set

$$(\langle z\rangle\vec{w})g = \begin{cases} w_{(fy)b}(z = 0)(g - fy) & \text{if } g(fy) = b \text{ for some } (y, b) \in \langle S\rangle, \\ (\langle x\rangle\vec{w}(z = 1))(g - fx) & \text{if } g(fx) = d \text{ and } fJ \subseteq \mathrm{def}(g), \\ \langle z'\rangle\, \vec{w}(g, z = z') & \text{otherwise,} \end{cases}$$

where in the last case z' is fresh w.r.t. the codomain of g. In particular, if $f = \mathbf{1}$, this definitions reads as

$$(\langle z\rangle\vec{w})g = \begin{cases} w_{yb}(z = 0)(g - y) & \text{if } gy = b \text{ for some } (y, b) \in \langle S\rangle, \\ (\langle x\rangle\vec{w}(z = 1))(g - x) & \text{if } gx = d \text{ and } J \subseteq \mathrm{def}(g), \\ \langle z'\rangle\, \vec{w}(g, z = z') & \text{otherwise.} \end{cases}$$

This definition deserves some explanation: in case $gy = b$ for $(y, b) \in \langle S\rangle$, $w_{yb} \in (A_{yb})_{s_z}$ and thus $w_{yb}(z = 0)(g - y) \in (A_{yb})_{(g-y)} = A_g$; in the second case where $gx = d$, $\vec{w}(z = 1)$ is an S-open box in $\vec{A}$, and thus, $\langle x\rangle\vec{w}(z = 1)$ is an an element of $(A_{xd})_{\mathbf{1}} = (|\mathbf{U}|_S\vec{A})_{\mathbf{1}}$ (cf. the definition in the previous lemma). It can be checked that this defines a (small) type $\mathbf{y}I \vdash [\mathbf{U}]_S\vec{A} = A$ satisfying (as types, not yet as Kan types)

$$\begin{aligned} ([\mathbf{U}]_S\vec{A})(y = b) &= A_{yb} && \text{for } (y, b) \in \langle S\rangle \\ ([\mathbf{U}]_S\vec{A})(x = d) &= |\mathbf{U}|_S\vec{A} \\ ([\mathbf{U}]_S\vec{A})f &= [\mathbf{U}]_{Sf}\vec{A}f && \text{if } f \text{ is defined on } x, J. \end{aligned}$$

The next step is to define the Kan structure on A_f where $f\colon I \to K$. W.l.o.g. we assume that $f = \mathbf{1}\colon I \to I$ as otherwise we use the above equations for the filling. Let $\vec{v}$ be an open box of shape $S' = ((x', d'); J'; I)$ in $A = [\mathbf{U}]_S\vec{A}$, i.e., $\vec{v}$ is given by adjacent-compatible $v_{yb} \in A_{(y=b)}$ for $(y, b) \in \langle S'\rangle$.

Note that if $(y,b) \in \langle S' \rangle \cap \langle S \rangle$, we have $v_{yb} \in A_{(y=b)} = (A_{yb})_{\mathbf{1}}$.

For $(y,b) \in \langle S' \rangle - \langle S \rangle$, we have $v_{yb} \in A_{(y=b)} = ([\mathbf{U}]_{S(y=b)}\vec{A}(y=b))_{\mathbf{1}}$, and so $v_{yb} = \langle z \rangle \vec{w}^{yb}$ where $\vec{w}^{yb}$ is an $S(y=b)s_z$ open box in $\vec{A}(y=b)s_z$ with the conditions described above. In particular, $w^{yb}_{\xi c} \in (A_{\xi c})_{(y=b)s_z}$ for $(\xi, c) \in \langle S \rangle$ with

$$w^{yb}_{\xi c}(\xi' = c') = w^{yb}_{\xi' c'}(\xi = c) \tag{4.11}$$

if also $(\xi', c') \in \langle S \rangle$ with $\xi \neq \xi'$. Moreover, since $\vec{v}$ is adjacent compatible we get that $v_{yb}(y' = b') = v_{y'b'}(y = b)$ for both $(y,b), (y',b') \in \langle S' \rangle - \langle S \rangle$, and thus the corresponding entries at position $(\xi, c) \in \langle S \rangle$ of the vectors $\vec{w}^{yb}$ and $\vec{w}^{y'b'}$ coincide, i.e.,

$$w^{yb}_{\xi c}(y' = b') = w^{y'b'}_{\xi c}(y = b). \tag{4.12}$$

Similar to the preceding lemma, for $(\xi, c) \in \langle S \rangle$ the adjacent-compatible family (not necessarily open box) of the $w^{yb}_{\xi c} \in (A_{\xi c})_{s_z(y=b)}$ for $y \in J' - (J, x)$ and $b \in \mathbf{2}$ is denoted by $\mathrm{aux}(\xi, c)$. Note that $\mathrm{aux}(\xi, c)$ is only defined on $J' - (J, x)$ and *not* on $\langle S' \rangle - \langle S \rangle$.

We want to construct $[A_{\mathbf{1}}]_{S'}\vec{v} = [([\mathbf{U}]_S \vec{A})_{\mathbf{1}}]_{S'}\vec{v} \in A_{\mathbf{1}}$ in a uniform way so that it satisfies for $f \colon I \to K$ defined on x', J'

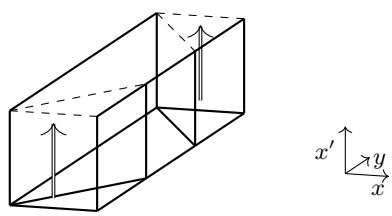
$$\big([([\mathbf{U}]_S \vec{A})_{\mathbf{1}}]_{S'}\vec{v}\big) f = [([\mathbf{U}]_S \vec{A})_f]_{S'f}(\vec{v}f)$$

that is,

$$\big([([\mathbf{U}]_S \vec{A})_{\mathbf{1}}]_{S'}\vec{v}\big) f = \begin{cases} [(A_{yb})_{(f-y)}]_{S'}(\vec{v}f) & \text{if } fy = b \text{ for some} \\ & (y,b) \in \{(x,d)\} \cup \langle S \rangle, \\ [([\mathbf{U}]_{Sf}\vec{A}f)_{\mathbf{1}}]_{S'f}(\vec{v}f) & \text{otherwise.} \end{cases} \tag{4.13}$$

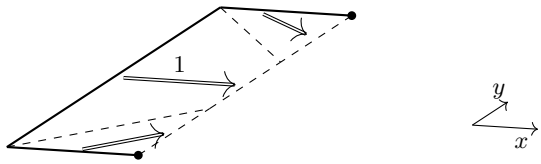
The element $[A_{\mathbf{1}}]_{S'}\vec{v}$ will be given by $\langle z \rangle \vec{w}$ with $\vec{w}$ an Ss_z open box in $\vec{A}s_z$ with the above provisos. The construction distinguishes several cases where in each case we assume that the previous cases didn't apply. We set $\bar{d} = 1 - d$ and $\bar{d}' = 1 - d'$.

1. *W.l.o.g.* $x, J \subseteq x', J'$. For each $y \in x, J$ with $y \notin x', J'$ we extend the input box $\vec{v}$ with $v_{yb} \in (A_{yb})_{s_z}$ $(b \in \mathbf{2})$ constructed as follows: v_{yb} is the filler in $(A_{yb})_{s_z}$ of the open box given by $\vec{v}(y = b)$ of shape $S'(y = b)$ in $(A_{yb})_{s_z}$. The final result will be an open box of shape $((x', d); J' \cup ((x, J) - (x', J')); I)$; to check that the so added sides indeed are adjacent compatible is similar to the verification done in the proof of the preceding lemma. Let us illustrate this in the special case with $J = y$, $J' = x$, and $x' \notin x, J$ (and $d = d' = 1$). We are given the sides enclosed by the solid lines and want to fill the whole shape:



In the picture we are hiding the extra dimension as discussed above. The input box is extended with the non-principal sides for y by filling the sides indicated with the double arrows.

2. *Case* $x = x'$ and $d = d'$. Then $(x, \bar{d}) \in \langle S' \rangle$. A simple special case like above but with $J = J' = y$ can be illustrated by:



Here the algorithm proceeds by first filling along the double arrow labeled "1", and then filling along the other arrows taking the side constructed in the first filling into account as a non-principal side (and where the opposing sides are the respective degenerates of the indicated points).

2.1. First, we construct $w_{x\bar{d}} \in (A_{x\bar{d}})_{s_z}$ by filling the open box

$$\begin{aligned} &v_{x\bar{d}} \in (A_{x\bar{d}})_{\mathbf{1}}, \\ &\mathrm{aux}(x, \bar{d}). \end{aligned}$$

Note that $\mathrm{aux}(x, \bar{d})$ is defined on the sides of $J' - (J, x)$, which is $J' - J$ in this case. Thus the open box has shape $((z, 1); J' - J; (I - x), z)$.

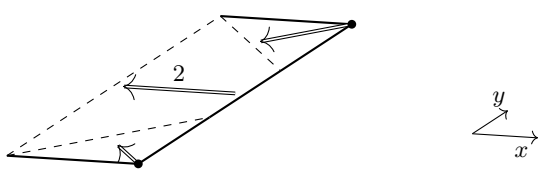
2.2. Next, we construct $w_{yb} \in (A_{yb})_{s_z}$ for $y \in J$ by filling the open box

$$\begin{aligned} &v_{yb} \in A_{(y=b)} = (A_{yb})_{\mathbf{1}}, && (4.14) \\ &v_{yb}(x = d)s_z \in (A_{yb})_{s_z(x=d)}, && (4.15) \\ &v_{x\bar{d}}(y = b) \in (A_{x\bar{d}})_{s_z(y=b)} = (A_{yb})_{s_z(x=\bar{d})} && (4.16) \\ &\mathrm{aux}(y, b) \end{aligned}$$

which is of shape $((z, 1); x, (J' - J); (I - y), z)$. Here (4.14) is the principal side, (4.15) is at the non-principal side (x, d), and (4.16) is at the $(x, \bar{d})$ side.

2.3. This concludes the construction of $\vec{w}$.

3. *Case* $x = x'$ and $d = 1 - d' = \bar{d'}$. Then the element $v_{xd} \in A_{(x=d)} = (A_{xd})_{\mathbf{1}} = (|\mathbf{U}|_S \vec{A})_{\mathbf{1}}$ is of the form $v_{xd} = \langle x \rangle \vec{u}$ where $\vec{u}$ is an S open box in $\vec{A}$ by definition of the compositions in the previous lemma. In the same special case as above, the situation can be depicted as:



Now the order of the filling is reversed: first fill the "triangles" as indicated, and then along the arrow labeled "2", taking into account the already constructed

sides.

3.1. First, for each $(y, b) \in J \times \mathbf{2}$ we construct w_{yb} by filling

$$\begin{aligned} & u_{yb}(x = d')s_z, && (4.17)\\ & v_{yb} \in A_{(y=b)} = (A_{yb})_{\mathbf{1}}, && (4.18)\\ & u_{yb} \in (A_{yb})_{\mathbf{1}}, && (4.19)\\ & \mathrm{aux}(y, b) \end{aligned}$$

in $(A_{yb})_{s_z}$ of shape $((x, \bar{d}); z, (J' - J); (I - y), z)$. Here, (4.17) is the principal face, and (4.18) and (4.19) are the non-principal faces at $(z, 0)$ and $(z, 1)$, respectively.

3.2. Next, we construct $w_{x\bar{d}} \in (A_{x\bar{d}})_{s_z}$ by filling the open box given by

$$\begin{aligned} & u_{x\bar{d}} \in (A_{x\bar{d}})_{\mathbf{1}}, \\ & w_{yb}(x = \bar{d}) \in (A_{yb})_{s_z(x=\bar{d})} && \text{for } (y, b) \in J \times \mathbf{2}, \\ & \mathrm{aux}(x, \bar{d}) && \text{at all sides of } J' - J \end{aligned}$$

of shape $((z, 0); J \cup (J' - J); (I - x), z) = ((z, 0); J'; (I - x), z)$.

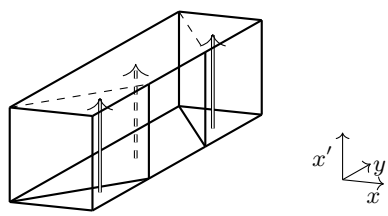
3.3. This concludes the construction of $\vec{w}$.

4. *Case* $x' \notin J$. As in the previous case, the element $v_{xd} \in A_{(x=d)} = (A_{xd})_{\mathbf{1}} = (|\mathbf{U}|_S \vec{A})_{\mathbf{1}}$ is of the form $v_{xd} = \langle x \rangle \vec{u}$ where $\vec{u}$ is an S-open box in $\vec{A}$. Moreover, $v_{x'\bar{d}'} \in A_{(x'=\bar{d}')} = ([\mathbf{U}]_S \vec{A})_{(x'=\bar{d}')}$ and since $x' \notin J$ and $x \neq x'$, so $(x', \bar{d}') \in \langle S' \rangle - \langle S \rangle$,

$$([\mathbf{U}]\vec{A})_{(x'=\bar{d}')} = ([\mathbf{U}]\vec{A}(x' = \bar{d}'))_{\mathbf{1}},$$

and so $v_{x'\bar{d}'} = \langle z \rangle \vec{w}^{x'\bar{d}'}$ with $\vec{w}^{x'\bar{d}'}$ an open box in $\vec{A}(x' = \bar{d}')$ (of shape $S(x' = \bar{d}')s_z$).

Let us again illustrate a special case with $J = y$ and $J' = x, y$. We are given those sides enclosed by solid lines and the top faces are missing:



Here the algorithm proceeds by first filling the middle cube along the dashed double arrow, and then the other two cubes double arrows taking into account the faces of the constructed middle cube (with opposing non-principle side

given by degenerates).

4.1. First, we construct $w_{x\bar{d}} \in (A_{x\bar{d}})_{s_z}$ by filling the open box

$w_{x\bar{d}}^{x'\bar{d}'} \in (A_{x\bar{d}})_{(x'=\bar{d}')}$	as principal side,
$v_{x\bar{d}} \in A_{(x=\bar{d})} = (A_{x\bar{d}})_{\mathbf{1}}$	at side $(z,0)$,
$u_{x\bar{d}} \in (A_{x\bar{d}})_{\mathbf{1}}$	at side $(z,1)$,
$\mathrm{aux}(x,\bar{d})$	at all sides of $J'-(x,J)$,

which is of shape $((x',d');z,(J'-(J,x));(I-x),z)$.

4.2. Next, we construct the other $w_{yb} \in (A_{yb})_{s_z}$ for $(y,b) \in J \times \mathbf{2}$ by filling

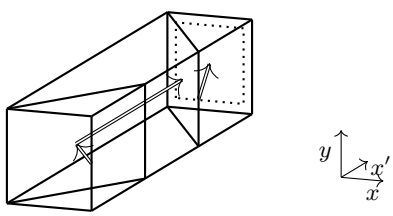
$w_{yb}^{x'\bar{d}'} \in (A_{yb})_{(x'=\bar{d}')}$	as principal side,
$v_{yb} \in A_{(y=b)} = (A_{yb})_{\mathbf{1}}$	at side $(z,0)$,
$u_{yb} \in (A_{yb})_{\mathbf{1}}$	at side $(z,1)$,
$w_{x\bar{d}}(y=b) \in (A_{x\bar{d}})_{s_z(y=b)} = (A_{yb})_{s_z(x=\bar{d})}$	at side $(x,\bar{d})$,
$u_{yb}(x=d)s_z \in (A_{yb})_{s_z(x=d)}$	at side (x,d),
$\mathrm{aux}(y,b)$	at all sides of $J'-(x,J)$

in $(A_{yb})_{s_z}$ which is of shape $((x',d');z,x,(J'-(x,J));(I-y),z)$.

4.3. This concludes the construction of $\vec{w}$.

5. *Case* $x' \in J$. As in the previous cases, the element $v_{xd} \in A_{(x=d)} = (A_{xd})_{\mathbf{1}} = (|\mathbf{U}|_S \vec{A})_{\mathbf{1}}$ is of the form $v_{xd} = \langle x \rangle \vec{u}$ where $\vec{u}$ is an S-open box in $\vec{A}$.

A special case (with $J = x'$ and $J' = x, y$) of this situation can depicted as a hollow box where the face indicated with dots is missing:



Here the algorithm proceeds in three steps: first filling the "prism" opposed to the dotted square (with principal face being the invisible degenerate square), then filling the middle cube, and finally filling the prism touching the dotted square.

5.1. First, we construct $w_{yb} \in (A_{yb})_{s_z}$ for $(y,b) \in \langle (x',d'); J - x' \rangle$ by filling the open box

$u_{yb}(x=d)s_z \in (A_{yb})_{s_z(x=d)}$	as principal side (x,d),
$v_{yb} \in A_{(y=b)} = (A_{yb})_{\mathbf{1}}$	at side $(z,0)$,
$u_{yb} \in (A_{yb})_{\mathbf{1}}$	at side $(z,1)$,
$\mathrm{aux}(y,b)$	at all sides of $J'-(x,J)$

of shape $((x,\bar{d});z,(J'-(x,J));(I-y),z)$.

5.2. Second, we construct $w_{x\bar{d}} \in (A_{x\bar{d}})_{s_z}$ by filling the following open box: the elements constructed so far induce an open box

$$\begin{aligned} w_{yb}(x=\bar{d}) \in (A_{yb})_{s_z(x=\bar{d})} &= (A_{x\bar{d}})_{s_z(y=b)} \\ \text{for } (y,b) &\in \langle (x',d');J-x';z,(I-(x,y))\rangle \end{aligned}$$

which we extend to an open box by adding the non-principal faces

$$\begin{array}{ll} v_{x\bar{d}} \in A_{(x=\bar{d})} = (A_{x\bar{d}})_{\mathbf{1}} & \text{at side } (z,0), \\ u_{yb} \in (A_{yb})_{\mathbf{1}} & \text{at side } (z,1), \\ \mathrm{aux}(y,b) & \text{at all sides of } J'-(x,J), \end{array}$$

to obtain an open box in $(A_{x\bar{d}})_{s_z}$ of shape $((x',d');(J-x'),z,(J'-J);(I-x),z)$.

5.3. Last, we construct the missing $w_{x'd'} \in (A_{x'd'})_{s_z}$ by filling the open box given by

$$\begin{array}{ll} u_{x'd'} \in (A_{x'd'})_{\mathbf{1}} & \text{as principal side at } (z,1), \\ w_{x\bar{d}}(x'=d') \in (A_{x\bar{d}})_{s_z(x'=d')} = (A_{x'd'})_{s_z(x=\bar{d})} & \text{at side } (x,\bar{d}), \\ u_{x'd'}(x=d)s_z & \text{at side } (x,d), \\ \mathrm{aux}(x',d') & \text{at all sides of } J'-(J,x), \\ w_{yb}(x'=d') \in (A_{yb})_{s_z(x'=d')} = (A_{x'd'})_{s_z(y=b)} & \text{for } (y,b) \text{ with } y \in J-x', \end{array}$$

which has shape $((z,0);x,(J'-(J,x)),(J-x');(I-x'),z)$.

5.4. This concludes the construction of $\vec{w}$.

The lengthy verification of the uniformity conditions is similar as sketched in the previous lemma and is omitted. □

Chapter 5

Conclusion

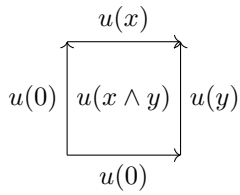
Let us conclude this part by summarizing what has been done and indicate future directions of research. We have given a model of dependent types based on a notion of cubical sets in a *constructive metatheory*. This model supports dependent products and sums, identity types, and universes. To give the interpretation of types we have to require a so-called uniform Kan structure which is a refinement of Kan's original extension condition on cubical sets; this condition is natural given the interpretation of the cubical set operations as "substitution operations".

One aspect not discussed in this part is the implementation [25] based on (a nominal variation of) the Kan cubical set model presented here. To sketch the basic idea, we start with type theory without identity types plus primitive notions such as `Id`, `refl`, `subst`, `substEq`, `isCenter`, and `funExt`. (The implementation also supports primitives which entail the univalence axiom.) There is an evaluation of terms (which may dependent on the just mentioned primitives) into values. Each value depends on finitely many names and there are the basic operations of cubical sets on values: we can rename a name into a fresh name or take a face. Values reflect the constructions in the model; e.g., there is path abstraction $\langle x \rangle u$ and filling operations. During evaluation names are introduced, e.g., $\texttt{refl}\, a$ is evaluated to $\langle x \rangle u$ where u is the value of a and x is a fresh name for u (in the implementation degeneracy maps are implicit). Moreover, the primitives are evaluated to values like it was described in this part of the thesis. Naturally, we also have to explain the Kan structure operations for each type. During type-checking the aforementioned primitives are treated as uninterpreted constants. But whenever the type-checker has to check for conversion of two terms this is done by *evaluating* their two values and comparing those values.

As we have seen, in the model the usual equation for the J-eliminator holds only up to propositional equality and *not* definitional equality. Restricting to regular Kan types (cf. Section 3.4) allows for definitional equality; this regularity is, however, not closed under function types. Andrew Swan [77] recently found a way to recover a definition of a proper identity type on top

of the weak identity type which satisfies the definitional equality; roughly speaking, elements in the proper identity types are built from elements in the weak one, so paths, but additionally are marked where they are known to be constant. In the next part of this thesis we will see a variation of the model where this construction has a particularly easy description.

The next part of this thesis is about a refined notion of the cubical set model covered in this part. The main difference to the present model is to use a variation of cubical sets: these are also equipped with the so-called *connections* which correspond to operations $x \wedge y$ and $x \vee y$ on names, satisfying the rules of a (bounded) distributive lattice; for cubical sets this allows, e.g., given a $u(x)$ depending on the name x to form the square (leaving degeneracy maps implicit):



This square is such that if u degenerate along x the square is (the degenerate of) $u(0)$. Moreover, this variation of cubical sets also allows taking *diagonals*, i.e., if a u depends on a name x there is an operation $u(x = y)$ even if u itself depends on y! Another difference to the present model is in the Kan structure: one only requires composition operations (the filling operations can be derived from those with the help of connections) but on more general "open box shapes". (One can also add symmetries $1 - x$ so that the structure on names becomes that of a *de Morgan algebra*.) This model has also been implemented and extended with an (in parts experimental) implementation of higher inductive types [27].

Another direction of future research is to explore the relations with the more categorically formulated model constructions using the notion of weak factorization systems. Since we don't model the definitional equality of `J` here, it is clear that the definition of `J` does not come from an underlying factorization system. Ongoing work [77] aims to give a variation of the current model using algebraic weak factorization system (also in a constructive metatheory). Indeed Swan and the author (private communication) could check that an argument by Christian Sattler (private communication) to give a model structure also works for cubical sets as considered in this part—assuming that one can show univalence for this model. Especially some aspects of the variation of the model mentioned in the previous paragraph has a lot of resemblance with the work on path object categories by van den Berg and Garner [81]; [33] shows that cubical sets with connections form an instance of such a path object category.

Another direction of work is to formulate a *cubical type theory*, i.e., a type theory where one can directly argue about and manipulate the (hyper) cube structure—this is done in the second part of this thesis. In such a theory,

names (and related concepts like name/path abstraction and application) are first-class entities, as well as the Kan structure is exposed to the users. Such a system formulates typing rules underlying the values from the implementations mentioned above. The uninterpreted constants from before, like function extensionality, can then directly be *implemented* inside this cubical type theory. Another aspect of directly being able to manipulate higher-dimensional cubes is that it allows for simpler proofs when doing synthetic homotopy theory inside type theory [56]. An implementation of such a cubical type theory is ongoing work [27]. A similar such "enriched" type theory for internalized parametricity was recently given in [12, 64] (based on a presheaf model similar to the one considered here). Similar type theories have been proposed by Altenkirch and Kaposi [5], Polonsky [73], and Brunerie and Licata [21].

Part II

Cubical Type Theory

Chapter 6

Cubical Type Theory: A Constructive Interpretation of the Univalence Axiom

6.1 Introduction

This work is a continuation of the program started in [15, 50] to provide a constructive justification of Voevodsky's univalence axiom [87]. This axiom allows many improvements for the formalization of mathematics in type theory: function extensionality, identification of isomorphic structures, etc. In order to preserve the good computational properties of type theory it is crucial that postulated constants have a computational interpretation. Like in [15, 50, 71] our work is based on a nominal extension of λ-calculus, using *names* to represent formally elements of the unit interval $[0, 1]$. This paper presents two main contributions.

The first one is a refinement of the semantics presented in [15, 50]. We add new operations on names corresponding to the fact that the interval $[0, 1]$ is canonically a de Morgan algebra [10]. This allows us to significantly simplify our semantical justifications. In the previous work, we noticed that it is crucial for the semantics of higher inductive types [79] to have a "diagonal" operation. By adding this operation we can provide a semantical justification of some higher inductive types and we give two examples (the spheres and propositional truncation). Another shortcoming of the previous work was that using path types as equality types did not provide a justification of the computation rule of the Martin-Löf identity type [59] as a judgmental equality. This problem has been solved by Andrew Swan [77], in the framework of [15, 50, 71], who showed that we can define a new type, *equivalent to*, but not judgmentally equal to the path type. This has a simple definition in the present framework.

The second contribution is the design of a type system[1] inspired by this semantics which extends Martin-Löf type theory [62, 59]. We add two new operations on contexts: addition of new names representing dimensions and a restriction operation. Using these we can define a notion of extensibility which generalizes the notion of being connected by a path, and then a Kan composition operation that expresses that being extensible is preserved along paths. We also define a new operation on types which expresses that this notion of extensibility is preserved by equivalences. The axiom of univalence, and composition for the universe, are then both expressible using this new operation.

The paper is organized as follows. The first part, Sections 6.2 to 6.7, presents the type system. The second part, Section 6.8, provides its semantics in cubical sets. Finally, in Section 6.9, we present two possible extensions: the addition of an identity type, and two examples of higher inductive types.

6.2 Basic Type Theory

In this section we introduce the version of dependent type theory on which the rest of the paper is based. This presentation is standard, but included for completeness. The type theory that we consider has a type of natural numbers, but no universes (we consider the addition of universes in Section 6.7). It also has β and η-conversion for dependent functions and surjective pairing for dependent pairs.

The syntax of contexts, terms and types is specified by:

$$
\begin{array}{llll}
\Gamma, \Delta & ::= & () \mid \Gamma, x : A & \text{Contexts} \\
\\
t, u, A, B & ::= & x \mid \lambda x : A.\, t \mid t\ u \mid (x : A) \to B & \Pi\text{-types} \\
& \mid & (t, u) \mid t.1 \mid t.2 \mid (x : A) \times B & \Sigma\text{-types} \\
& \mid & \mathsf{0} \mid \mathsf{s}\ u \mid \mathsf{natrec}\ t\ u \mid \mathsf{N} & \text{Natural numbers}
\end{array}
$$

We write $A \to B$ for the non-dependent function space and $A \times B$ for the type of non-dependent pairs. Terms and types are considered up to α-equivalence of bound variables. Substitutions, written $\sigma = (x_1/u_1, \ldots, x_n/u_n)$, are defined to act on expressions as usual, i.e., simultaneously replacing x_i by u_i, renaming bound variables whenever necessary. The inference rules of this system are presented in Figure 6.1 and Figure 6.2 where in the η-rule for Π- and Σ-types we omitted the premises that t and u should have the respective type.

We define $\Delta \vdash \sigma : \Gamma$ by induction on Γ. We have $\Delta \vdash () : ()$ (empty substitution) and $\Delta \vdash (\sigma, x/u) : \Gamma, x : A$ if $\Delta \vdash \sigma : \Gamma$ and $\Delta \vdash u : A\sigma$.

We write J for an arbitrary judgment and, as usual, we consider also *hypothetical* judgments $\Gamma \vdash \mathsf{J}$ in a *context* Γ.

[1] We have implemented a type-checker for this system in Haskell, which is available at: `https://github.com/mortberg/cubicaltt`

Well-formed contexts, $\Gamma \vdash$ (The condition $x \notin \mathrm{dom}(\Gamma)$ means that x is not declared in Γ)

$$\frac{}{() \vdash} \qquad \frac{\Gamma \vdash A}{\Gamma, x : A \vdash} \quad (x \notin \mathrm{dom}(\Gamma))$$

Well-formed types, $\Gamma \vdash A$

$$\frac{\Gamma, x : A \vdash B}{\Gamma \vdash (x : A) \to B} \qquad \frac{\Gamma, x : A \vdash B}{\Gamma \vdash (x : A) \times B} \qquad \frac{\Gamma \vdash}{\Gamma \vdash \mathsf{N}}$$

Well-typed terms, $\Gamma \vdash t : A$

$$\frac{\Gamma \vdash t : A \qquad \Gamma \vdash A = B}{\Gamma \vdash t : B} \qquad \frac{\Gamma, x : A \vdash t : B}{\Gamma \vdash \lambda x : A.\, t : (x : A) \to B}$$

$$\frac{\Gamma \vdash}{\Gamma \vdash x : A}\ (x : A \in \Gamma) \qquad \frac{\Gamma \vdash t : (x : A) \to B \qquad \Gamma \vdash u : A}{\Gamma \vdash t\ u : B(x/u)}$$

$$\frac{\Gamma \vdash t : (x : A) \times B}{\Gamma \vdash t.1 : A} \qquad \frac{\Gamma \vdash t : (x : A) \times B}{\Gamma \vdash t.2 : B(x/t.1)}$$

$$\frac{\Gamma, x : A \vdash B \qquad \Gamma \vdash t : A \qquad \Gamma \vdash u : B(x/t)}{\Gamma \vdash (t, u) : (x : A) \times B} \qquad \frac{\Gamma \vdash}{\Gamma \vdash \mathsf{0} : \mathsf{N}} \qquad \frac{\Gamma \vdash n : \mathsf{N}}{\Gamma \vdash \mathsf{s}\ n : \mathsf{N}}$$

$$\frac{\Gamma, x : \mathsf{N} \vdash P \qquad \Gamma \vdash a : P(x/\mathsf{0}) \qquad \Gamma \vdash b : (n : \mathsf{N}) \to P(x/n) \to P(x/\mathsf{s}\ n)}{\Gamma \vdash \mathsf{natrec}\ a\ b : (x : \mathsf{N}) \to P}$$

Figure 6.1: Context, type, and term formation rules of the basic type theory

$\boxed{\text{Type equality, } \Gamma \vdash A = B}$ (Congruence and equivalence rules which are omitted)

$\boxed{\text{Term equality, } \Gamma \vdash a = b : A}$ (Congruence and equivalence rules are omitted)

$$\frac{\Gamma \vdash t = u : A \qquad \Gamma \vdash A = B}{\Gamma \vdash t = u : B} \qquad \frac{\Gamma, x : A \vdash t : B \qquad \Gamma \vdash u : A}{\Gamma \vdash (\lambda x : A.\ t)\ u = t(x/u) : B(x/u)}$$

$$\frac{\Gamma, x : A \vdash t\ x = u\ x : B}{\Gamma \vdash t = u : (x : A) \to B} \qquad \frac{\Gamma, x : A \vdash B \qquad \Gamma \vdash t : A \qquad \Gamma \vdash u : B(x/t)}{\Gamma \vdash (t, u).1 = t : A}$$

$$\frac{\Gamma, x : A \vdash B \qquad \Gamma \vdash t : A \qquad \Gamma \vdash u : B(x/t)}{\Gamma \vdash (t, u).2 = u : B(x/t)}$$

$$\frac{\Gamma, x : A \vdash B \qquad \Gamma \vdash t.1 = u.1 : A \qquad \Gamma \vdash t.2 = u.2 : B(x/t.1)}{\Gamma \vdash t = u : (x : A) \times B}$$

$$\frac{\Gamma, x : \mathsf{N} \vdash P \qquad \Gamma \vdash a : P(x/\mathsf{0}) \qquad \Gamma \vdash b : (n : \mathsf{N}) \to P(x/n) \to P(x/\mathsf{s}\ n)}{\Gamma \vdash \mathsf{natrec}\ a\ b\ \mathsf{0} = a : P(x/\mathsf{0})}$$

$$\frac{\begin{array}{c}\Gamma, x : \mathsf{N} \vdash P \\ \Gamma \vdash a : P(x/\mathsf{0}) \qquad \Gamma \vdash b : (n : \mathsf{N}) \to P(x/n) \to P(x/\mathsf{s}\ n) \qquad \Gamma \vdash n : \mathsf{N}\end{array}}{\Gamma \vdash \mathsf{natrec}\ a\ b\ (\mathsf{s}\ n) = b\ n\ (\mathsf{natrec}\ a\ b\ n) : P(x/\mathsf{s}\ n)}$$

Figure 6.2: Judgmental equality rules of the basic type theory

The following lemma will be valid for all extensions of type theory we consider below.

Lemma 6.2.1. *Substitution is admissible:*

$$\frac{\Gamma \vdash \mathsf{J} \qquad \Delta \vdash \sigma : \Gamma}{\Delta \vdash \mathsf{J}\sigma}$$

In particular, weakening is admissible, i.e., a judgment valid in a context stays valid in any extension of this context.

6.3 Path Types

As in [15, 71] we assume that we are given a discrete infinite set of names (representing directions) $i, j, k, \ldots$ We define $\mathbb{I}$ to be the free de Morgan algebra [10] on this set of names. This means that $\mathbb{I}$ is a bounded distributive lattice with top element 1 and bottom element 0 with an involution $1 - r$ satisfying:

$$1 - 0 = 1 \qquad 1 - 1 = 0 \qquad 1 - (r \vee s) = (1 - r) \wedge (1 - s)$$

$$1 - (r \wedge s) = (1 - r) \vee (1 - s)$$

The elements of $\mathbb{I}$ can hence be described by the following grammar:

$$r, s \quad ::= \quad 0 \mid 1 \mid i \mid 1 - r \mid r \wedge s \mid r \vee s$$

The set $\mathbb{I}$ also has decidable equality, and as a distributive lattice, it can be described as the free distributive lattice generated by symbols i and $1 - i$ [10]. As in [15], the elements in $\mathbb{I}$ can be thought as formal representations of elements in $[0, 1]$, with $r \wedge s$ representing $min(r, s)$ and $r \vee s$ representing $max(r, s)$. With this in mind it is clear that $(1 - r) \wedge r \neq 0$ and $(1 - r) \vee r \neq 1$ (unless r is 0 or 1).

Remark 6.3.1. We could instead also use a so-called Kleene algebra [52], i.e., a de Morgan algebra satisfying in addition $r \wedge (1-r) \leqslant s \vee (1-s)$. The free Kleene algebra on the set of names can be described as above but by additionally imposing the equations $i \wedge (1 - i) \leqslant j \vee (1 - j)$ on the generators; this still has a decidable equality. Note that $[0, 1]$ with the operations described above is a Kleene algebra. With this added condition, $r = s$ if, and only if, their interpretations in $[0, 1]$ are equal. A consequence of using a Kleene algebra instead would be that more terms would be judgmentally equal in the type theory.

6.3.1 Syntax and Inference Rules

Contexts can now be extended with name declarations:

$$\Gamma, \Delta \quad ::= \quad \ldots \mid \Gamma, i : \mathbb{I}$$

together with the context rule:

$$\frac{\Gamma \vdash}{\Gamma, i : \mathbb{I} \vdash} \; (i \notin \mathrm{dom}(\Gamma))$$

A judgment of the form $\Gamma \vdash r : \mathbb{I}$ means that $\Gamma \vdash$ and r in $\mathbb{I}$ depends only on the names declared in Γ. The judgment $\Gamma \vdash r = s : \mathbb{I}$ means that r and s are equal as elements of $\mathbb{I}$, $\Gamma \vdash r : \mathbb{I}$, and $\Gamma \vdash s : \mathbb{I}$. Note, that judgmental equality for $\mathbb{I}$ will be re-defined once we introduce restricted contexts in Section 6.4.

The extension to the syntax of basic dependent type theory is:

$$\begin{array}{rcll} t, u, A, B & ::= & \ldots & \\ & | & \mathsf{Path}\ A\ t\ u \ \mid\ \langle i \rangle\ t \ \mid\ t\ r & \text{Path types} \end{array}$$

Path abstraction, $\langle i \rangle\ t$, binds the name i in t, and path application, $t\ r$, applies a term t to an element $r : \mathbb{I}$. This is similar to the notion of name-abstraction in nominal sets [70].

The substitution operation now has to be extended to substitutions of the form (i/r). There are special substitutions of the form $(i/0)$ and $(i/1)$ corresponding to taking faces of an n-dimensional cube, we write these simply as $(i0)$ and $(i1)$.

The inference rules for path types are presented in Figure 6.3 where again in the η-rule we omitted that t and u should be appropriately typed.

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash t : A \qquad \Gamma \vdash u : A}{\Gamma \vdash \mathsf{Path}\ A\ t\ u} \qquad \frac{\Gamma \vdash A \qquad \Gamma, i : \mathbb{I} \vdash t : A}{\Gamma \vdash \langle i \rangle\ t : \mathsf{Path}\ A\ t(i0)\ t(i1)}$$

$$\frac{\Gamma \vdash t : \mathsf{Path}\ A\ u_0\ u_1 \qquad \Gamma \vdash r : \mathbb{I}}{\Gamma \vdash t\ r : A}$$

$$\frac{\Gamma \vdash A \qquad \Gamma, i : \mathbb{I} \vdash t : A \qquad \Gamma \vdash r : \mathbb{I}}{\Gamma \vdash (\langle i \rangle\ t)\ r = t(i/r) : A} \qquad \frac{\Gamma, i : \mathbb{I} \vdash t\ i = u\ i : A}{\Gamma \vdash t = u : \mathsf{Path}\ A\ u_0\ u_1}$$

$$\frac{\Gamma \vdash t : \mathsf{Path}\ A\ u_0\ u_1}{\Gamma \vdash t\ 0 = u_0 : A} \qquad \frac{\Gamma \vdash t : \mathsf{Path}\ A\ u_0\ u_1}{\Gamma \vdash t\ 1 = u_1 : A}$$

Figure 6.3: Inference rules for path types

We define $1_a : \mathsf{Path}\ A\ a\ a$ as $1_a = \langle i \rangle\ a$, which corresponds to a proof of reflexivity.

The intuition is that a type in a context with n names corresponds to an

n-dimensional cube:

$() \vdash A$	$\bullet\ A$
$i : \mathbb{I} \vdash A$	$A(i0) \xrightarrow{A} A(i1)$
$i : \mathbb{I}, j : \mathbb{I} \vdash A$	$\begin{array}{ccc} A(i0)(j1) & \xrightarrow{A(j1)} & A(i1)(j1) \\ {\scriptstyle A(i0)}\uparrow & A & \uparrow{\scriptstyle A(i1)} \\ A(i0)(j0) & \xrightarrow[A(j0)]{} & A(i1)(j0) \end{array}$
$\vdots$	$\vdots$

Note that $A(i0)(j0) = A(j0)(i0)$. The substitution (i/j) corresponds to renaming a dimension, while $(i/1-i)$ corresponds to the inversion of a path. If we have $i : \mathbb{I} \vdash p$ with $p(i0) = a$ and $p(i1) = b$ then it can be seen as a line

$$a \xrightarrow{p} b$$

in direction i, then:

$$b \xrightarrow{p(i/1-i)} a$$

The substitutions $(i/i \wedge j)$ and $(i/i \vee j)$ correspond to special kinds of degeneracies called *connections* [19]. The connections $p(i/i \wedge j)$ and $p(i/i \vee j)$ can be drawn as the squares:

$$\begin{array}{ccc} a & \xrightarrow{p} & b \\ {\scriptstyle p(i0)}\uparrow & p(i/i \wedge j) & \uparrow{\scriptstyle p(i/j)} \\ a & \xrightarrow[p(i0)]{} & a \end{array} \qquad \begin{array}{ccc} b & \xrightarrow{p(i1)} & b \\ {\scriptstyle p(i/j)}\uparrow & p(i/i \vee j) & \uparrow{\scriptstyle p(i1)} \\ a & \xrightarrow[p]{} & b \end{array} \qquad \begin{array}{ccc} j & \uparrow & \\ & \rightarrow & i \end{array}$$

where, for instance, the right-hand side of the left square is computed as

$$p(i/i \wedge j)(i1) \quad = \quad p(i/1 \wedge j) \quad = \quad p(i/j)$$

and the bottom and left-hand sides are degenerate.

6.3.2 Examples

Representing equalities using path types allows novel definitions of many standard operations on identity types that are usually proved by identity elimination. For instance, the fact that the images of two equal elements are equal can be defined as:

$$\frac{\Gamma \vdash a : A \qquad \Gamma \vdash b : A \qquad \Gamma \vdash f : A \to B \qquad \Gamma \vdash p : \mathsf{Path}\ A\ a\ b}{\Gamma \vdash \langle i \rangle\ f\ (p\ i) : \mathsf{Path}\ B\ (f\ a)\ (f\ b)}$$

This operation satisfies some judgmental equalities that do not hold judgmentally when the identity type is defined as an inductive family (see Section 7.2 of [15] for details).

We can also define new operations, for instance, function extensionality for path types can be proved as:

$$\frac{\Gamma \vdash f : (x : A) \to B \qquad \Gamma \vdash g : (x : A) \to B \qquad \Gamma \vdash p : (x : A) \to \mathsf{Path}\ B\ (f\ x)\ (g\ x)}{\Gamma \vdash \langle i \rangle\ \lambda x : A.\ p\ x\ i : \mathsf{Path}\ ((x : A) \to B)\ f\ g}$$

To see that this is correct we check that the term has the correct faces, for instance:

$$(\langle i \rangle\ \lambda x : A.\ p\ x\ i)\ 0 = \lambda x : A.\ p\ x\ 0 = \lambda x : A.\ f\ x = f$$

We can also justify the fact that singletons are contractible, that is, that any element in $(x : A) \times (\mathsf{Path}\ A\ a\ x)$ is equal to $(a, 1_a)$:

$$\frac{\Gamma \vdash p : \mathsf{Path}\ A\ a\ b}{\Gamma \vdash \langle i \rangle\ (p\ i, \langle j \rangle\ p\ (i \wedge j)) : \mathsf{Path}\ ((x : A) \times (\mathsf{Path}\ A\ a\ x))\ (a, 1_a)\ (b, p)}$$

As in the previous work [15, 50] we need to add *composition operations*, defined by induction on the type, in order to justify the elimination principle for paths.

6.4 Systems, Composition, and Transport

In this section we define the operation of context *restriction* which will allow us to describe new geometrical shapes corresponding to "sub-polyhedra" of a cube. Using this we can define the composition operation. From this operation we will also be able to define the transport operation and the elimination principle for Path types.

6.4.1 The Face Lattice

The *face lattice*, $\mathbb{F}$, is the distributive lattice generated by symbols $(i = 0)$ and $(i = 1)$ with the relation $(i = 0) \wedge (i = 1) = 0_{\mathbb{F}}$. The elements of the face lattice, called *face formulas*, can be described by the grammar:

$$\varphi, \psi \quad ::= \quad 0_{\mathbb{F}} \ \mid\ 1_{\mathbb{F}} \ \mid\ (i = 0) \ \mid\ (i = 1) \ \mid\ \varphi \wedge \psi \ \mid\ \varphi \vee \psi$$

There is a canonical lattice map $\mathbb{I} \to \mathbb{F}$ sending i to $(i = 1)$ and $1 - i$ to $(i = 0)$. We write $(r = 1)$ for the image of $r : \mathbb{I}$ in $\mathbb{F}$ and we write $(r = 0)$ for $(1 - r = 1)$. We have $(r = 1) \wedge (r = 0) = 0_{\mathbb{F}}$ and we define the lattice map $\mathbb{F} \to \mathbb{F},\ \psi \longmapsto \psi(i/r)$ sending $(i = 1)$ to $(r = 1)$ and $(i = 0)$ to $(r = 0)$.

Any element of $\mathbb{F}$ is the join of the irreducible elements below it. An irreducible element of this lattice is a *face*, i.e., a conjunction of elements of

the form $(i = 0)$ and $(j = 1)$. This provides a disjunctive normal form for face formulas, and it follows from this that the equality on $\mathbb{F}$ is decidable.

Geometrically, the face formulas describe "sub-polyhedra" of a cube. For instance, the element $(i = 0) \vee (j = 1)$ can be seen as the union of two faces of the square in directions j and i. If I is a finite set of names, we define the *boundary* of I as the element ∂_I of $\mathbb{F}$ which is the disjunction of all $(i = 0) \vee (i = 1)$ for i in I. It is the greatest element depending at most on elements in I which is $< 1_{\mathbb{F}}$.

We write $\Gamma \vdash \psi : \mathbb{F}$ to mean that ψ is a face formula using only the names declared in Γ. We introduce then the new *restriction* operation on contexts:

$$\Gamma, \Delta \quad ::= \quad \ldots \quad | \quad \Gamma, \varphi$$

together with the rule:

$$\frac{\Gamma \vdash \varphi : \mathbb{F}}{\Gamma, \varphi \vdash}$$

This allows us to describe new geometrical shapes: as we have seen above, a type in a context $\Gamma = i : \mathbb{I}, j : \mathbb{I}$ can be thought of as a square, and a type in the restricted context Γ, φ will then represent a compatible union of faces of this square. This can be illustrated by:

$i : \mathbb{I}, (i = 0) \vee (i = 1) \vdash A$	$A(i0) \bullet \qquad\qquad \bullet\, A(i1)$
$i : \mathbb{I}, j : \mathbb{I}, (i = 0) \vee (j = 1) \vdash A$	$A(i0)(j1) \xrightarrow{A(j1)} A(i1)(j1)$ $A(i0) \uparrow$ $A(i0)(j0)$
$i : \mathbb{I}, j : \mathbb{I}, (i = 0) \vee (i = 1) \vee (j = 0) \vdash A$	$A(i0)(j1) \qquad A(i1)(j1)$ $A(i0) \uparrow \qquad\qquad \uparrow A(i1)$ $A(i0)(j0) \xrightarrow[A(j0)]{} A(i1)(j0)$

There is a canonical map from the lattice $\mathbb{F}$ to the congruence lattice of $\mathbb{I}$, which is distributive [10], sending $(i = 1)$ to the congruence identifying i with 1 (and $1 - i$ with 0) and sending $(i = 0)$ to the congruence identifying i with 0 (and $1 - i$ with 1). In this way, any element ψ of $\mathbb{F}$ defines a congruence $r = s$ (mod. ψ) on $\mathbb{I}$.

This congruence can be described as a substitution if ψ is irreducible; for instance, if ψ is $(i = 0) \wedge (j = 1)$ then $r = s$ (mod. ψ) is equivalent to $r(i0)(j1) = s(i0)(j1)$. The congruence associated to $\psi = \varphi_0 \vee \varphi_1$ is the meet

of the congruences associated to φ_0 and φ_1 respectively, so that we have, e.g., $i = 1 - j$ (mod. ψ) if $\varphi_0 = (i = 0) \wedge (j = 1)$ and $\varphi_1 = (i = 1) \wedge (j = 0)$.

To any context Γ we can associate recursively a congruence on $\mathbb{I}$, the congruence on Γ, ψ being the join of the congruence defined by Γ and the congruence defined by ψ. The congruence defined by () is equality in $\mathbb{I}$, and an extension $x : A$ or $i : \mathbb{I}$ does not change the congruence. The judgment $\Gamma \vdash r = s : \mathbb{I}$ then means that $r = s$ (mod. Γ), $\Gamma \vdash r : \mathbb{I}$, and $\Gamma \vdash s : \mathbb{I}$.

In the case where Γ does not use the restriction operation, this judgment means $r = s$ in $\mathbb{I}$. If i is declared in Γ, then $\Gamma, (i = 0) \vdash r = s : \mathbb{I}$ is equivalent to $\Gamma \vdash r(i0) = s(i0) : \mathbb{I}$. Similarly any context Γ defines a congruence on $\mathbb{F}$ with $\Gamma, \psi \vdash \varphi_0 = \varphi_1 : \mathbb{F}$ being equivalent to $\Gamma \vdash \psi \wedge \varphi_0 = \psi \wedge \varphi_1 : \mathbb{F}$.

As explained above, the elements of $\mathbb{I}$ can be seen as formal representations of elements in the interval $[0, 1]$. The elements of $\mathbb{F}$ can then be seen as formulas on elements of $[0, 1]$. We have a simple form of *quantifier elimination* on $\mathbb{F}$: given a name i, we define $\forall i \colon \mathbb{F} \to \mathbb{F}$ as the lattice morphism sending $(i = 0)$ and $(i = 1)$ to $0_{\mathbb{F}}$, and being the identity on all the other generators. If ψ is independent of i, we have $\psi \leqslant \varphi$ if, and only if, $\psi \leqslant \forall i.\varphi$. For example, if φ is $(i = 0) \vee ((i = 1) \wedge (j = 0)) \vee (j = 1)$, then $\forall i.\varphi$ is $(j = 1)$. This operation will play a crucial role in Section 6.6.2 for the definition of composition of glueing.

Since $\mathbb{F}$ is not a Boolean algebra, we don't have in general $\varphi = (\varphi \wedge (i = 0)) \vee (\varphi \wedge (i = 1))$, but we always have the following decomposition:

Lemma 6.4.1. *For any element φ of $\mathbb{F}$ and any name i we have*

$$\varphi = (\forall i.\varphi) \vee (\varphi \wedge (i = 0)) \vee (\varphi \wedge (i = 1))$$

We also have $\varphi \wedge (i = 0) \leqslant \varphi(i0)$ and $\varphi \wedge (i = 1) \leqslant \varphi(i1)$.

6.4.2 Syntax and Inference Rules for Systems

Systems allow to introduce "sub-polyhedra" as compatible unions of cubes. The extension to the syntax of dependent type theory with path types is:

$$\begin{array}{rcll} t, u, A, B & ::= & \ldots & \\ & | & [\ \varphi_1\ t_1, \ldots, \varphi_n\ t_n\ ] & \text{Systems} \end{array}$$

We allow $n = 0$ and get the empty system $[\ ]$. As explained above, a context now corresponds in general to the union of sub-faces of a cube. In Figure 6.4 we provide operations for combining compatible systems of types and elements, the side condition for these rules is that $\Gamma \vdash \varphi_1 \vee \cdots \vee \varphi_n = 1_{\mathbb{F}} : \mathbb{F}$. This condition requires Γ to be sufficiently restricted: for example $\Delta, (i = 0) \vee (i = 1) \vdash (i = 0) \vee (i = 1) = 1_{\mathbb{F}}$. The first rule introduces systems of types, each defined on one φ_i and requiring the types to agree whenever they overlap; the second rule is the analogous rule for terms. The last two rules make sure that systems have the correct faces. The third inference rule says that that any judgment which is valid locally at each φ_i is valid; note that in particular $n = 0$ is allowed (then the side condition becomes $\Gamma \vdash 0_{\mathbb{F}} = 1_{\mathbb{F}} : \mathbb{F}$).

$$\frac{\Gamma, \varphi_1 \vdash A_1 \quad \cdots \quad \Gamma, \varphi_n \vdash A_n \qquad \Gamma, \varphi_i \wedge \varphi_j \vdash A_i = A_j \quad (1 \leqslant i, j \leqslant n)}{\Gamma \vdash [\ \varphi_1\ A_1, \ldots, \varphi_n\ A_n\ ]}$$

$$\frac{\Gamma \vdash A \qquad \Gamma, \varphi_1 \vdash t_1 : A \quad \cdots \quad \Gamma, \varphi_n \vdash t_n : A \qquad \Gamma, \varphi_i \wedge \varphi_j \vdash t_i = t_j : A \ \ (1 \leqslant i, j \leqslant n)}{\Gamma \vdash [\ \varphi_1\ t_1, \ldots, \varphi_n\ t_n\ ] : A}$$

$$\frac{\Gamma, \varphi_1 \vdash \mathsf{J} \quad \cdots \quad \Gamma, \varphi_n \vdash \mathsf{J}}{\Gamma \vdash \mathsf{J}}$$

$$\frac{\Gamma \vdash [\ \varphi_1\ A_1, \ldots, \varphi_n\ A_n\ ] \qquad \Gamma \vdash \varphi_i = 1_{\mathbb{F}} : \mathbb{F}}{\Gamma \vdash [\ \varphi_1\ A_1, \ldots, \varphi_n\ A_n\ ] = A_i}$$

$$\frac{\Gamma \vdash [\ \varphi_1\ t_1, \ldots, \varphi_n\ t_n\ ] : A \qquad \Gamma \vdash \varphi_i = 1_{\mathbb{F}} : \mathbb{F}}{\Gamma \vdash [\ \varphi_1\ t_1, \ldots, \varphi_n\ t_n\ ] = t_i : A}$$

Figure 6.4: Inference rules for systems with side condition $\Gamma \vdash \varphi_1 \vee \cdots \vee \varphi_n = 1_{\mathbb{F}} : \mathbb{F}$

Note that when $n = 0$ the second of the above rules should be read as: if $\Gamma \vdash 0_{\mathbb{F}} = 1_{\mathbb{F}} : \mathbb{F}$ and $\Gamma \vdash A$, then $\Gamma \vdash [\,] : A$.

We extend the definition of the substitution judgment by $\Delta \vdash \sigma : \Gamma, \varphi$ if $\Delta \vdash \sigma : \Gamma$, $\Gamma \vdash \varphi : \mathbb{F}$, and $\Delta \vdash \varphi\sigma = 1_{\mathbb{F}} : \mathbb{F}$.

If $\Gamma, \varphi \vdash u : A$, then $\Gamma \vdash a : A[\varphi \mapsto u]$ is an abbreviation for $\Gamma \vdash a : A$ and $\Gamma, \varphi \vdash a = u : A$. In this case, we see this element a as a witness that the partial element u, defined on the "extent" φ (using the terminology from [36]), is *extensible*. More generally, we write $\Gamma \vdash a : A[\varphi_1 \mapsto u_1, \ldots, \varphi_k \mapsto u_k]$ for $\Gamma \vdash a : A$ and $\Gamma, \varphi_i \vdash a = u_i : A$ for $i = 1, \ldots, k$.

For instance, if $\Gamma, i : \mathbb{I} \vdash A$ and $\Gamma, i : \mathbb{I}, \varphi \vdash u : A$ where $\varphi = (i = 0) \vee (i = 1)$ then the element u is determined by two elements $\Gamma \vdash a_0 : A(i0)$ and $\Gamma \vdash a_1 : A(i1)$ and an element $\Gamma, i : \mathbb{I} \vdash a : A[(i = 0) \mapsto a_0, (i = 1) \mapsto a_1]$ gives a path connecting a_0 and a_1.

Lemma 6.4.2. *The following rules are admissible:*[2]

$$\frac{\Gamma \vdash \varphi \leqslant \psi : \mathbb{F} \qquad \Gamma, \psi \vdash \mathsf{J}}{\Gamma, \varphi \vdash \mathsf{J}} \qquad \frac{\Gamma, 1_{\mathbb{F}} \vdash \mathsf{J}}{\Gamma \vdash \mathsf{J}} \qquad \frac{\Gamma, \varphi, \psi \vdash \mathsf{J}}{\Gamma, \varphi \wedge \psi \vdash \mathsf{J}}$$

Furthermore, if φ is independent of i, the following rules are admissible

$$\frac{\Gamma, i : \mathbb{I}, \varphi \vdash \mathsf{J}}{\Gamma, \varphi, i : \mathbb{I} \vdash \mathsf{J}}$$

[2]The inference rules with double line are each a pair of rules, because they can be used in both directions.

and it follows that we have in general:

$$\frac{\Gamma, i : \mathbb{I}, \varphi \vdash \mathsf{J}}{\Gamma, \forall i.\varphi, i : \mathbb{I} \vdash \mathsf{J}}$$

6.4.3 Composition Operation

The syntax of compositions is given by:

$$\begin{array}{lcll} t, u, A, B & ::= & \ldots & \\ & | & \mathsf{comp}^i\ A\ [\varphi \mapsto u]\ a_0 & \text{Compositions} \end{array}$$

where u is a system on the extent φ.

The composition operation expresses that being extensible is preserved along paths: if a partial path is extensible at 0, then it is extensible at 1.

$$\frac{\Gamma \vdash \varphi : \mathbb{F} \qquad \Gamma, i : \mathbb{I} \vdash A \qquad \Gamma, \varphi, i : \mathbb{I} \vdash u : A \qquad \Gamma \vdash a_0 : A(i0)[\varphi \mapsto u(i0)]}{\Gamma \vdash \mathsf{comp}^i\ A\ [\varphi \mapsto u]\ a_0 : A(i1)[\varphi \mapsto u(i1)]}$$

Note that comp^i binds i in A and u and that we have in particular the following equality judgments for systems:

$$\Gamma \vdash \mathsf{comp}^i\ A\ [1_{\mathbb{F}} \mapsto u]\ a_0 = u(i1) : A(i1)$$

If we have a substitution $\Delta \vdash \sigma : \Gamma$, then

$$(\mathsf{comp}^i\ A\ [\varphi \mapsto u]\ a_0)\sigma = \mathsf{comp}^j\ A(\sigma, i/j)\ [\varphi\sigma \mapsto u(\sigma, i/j)]\ a_0\sigma$$

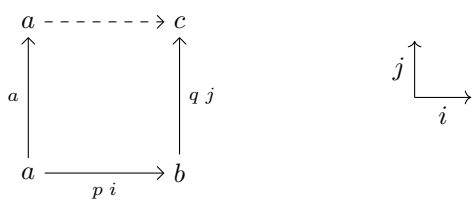
where j is fresh for Δ, which corresponds semantically to the *uniformity* [15, 50] of the composition operation.

We abbreviate $[\bigvee_i \varphi_i \mapsto [\varphi_1\ u_1, \ldots, \varphi_n\ u_n]]$ by $[\varphi_1 \mapsto u_1, \ldots, \varphi_n \mapsto u_n]$, and in particular we write $[]$ for $[0_{\mathbb{F}} \mapsto [\,]]$.

Example 6.4.3. With composition we can justify transitivity of path types:

$$\frac{\Gamma \vdash p : \mathsf{Path}\ A\ a\ b \qquad \Gamma \vdash q : \mathsf{Path}\ A\ b\ c}{\Gamma \vdash \langle i \rangle\ \mathsf{comp}^j\ A\ [(i = 0) \mapsto a, (i = 1) \mapsto q\ j]\ (p\ i) : \mathsf{Path}\ A\ a\ c}$$

This composition can be visualized as the dashed arrow in the square:



6.4.4 Kan Filling Operation

As we have connections we also get Kan filling operations from compositions:

$$\Gamma, i : \mathbb{I} \vdash \mathsf{fill}^i\ A\ [\varphi \mapsto u]\ a_0 = \\ \mathsf{comp}^j\ A(i/i \wedge j)\ [\varphi \mapsto u(i/i \wedge j), (i = 0) \mapsto a_0]\ a_0 : A$$

where j is fresh for Γ. The element $\Gamma, i : \mathbb{I} \vdash v = \mathsf{fill}^i\ A\ [\varphi \mapsto u]\ a_0 : A$ satisfies:

$$\Gamma \vdash v(i0) = a_0 : A(i0) \qquad \Gamma \vdash v(i1) = \mathsf{comp}^i\ A\ [\varphi \mapsto u]\ a_0 : A(i1)$$

$$\Gamma, \varphi, i : \mathbb{I} \vdash v = u : A$$

This means that we can not only compute the lid of an open box but also its filling. If φ is the boundary formula on the names declared in Γ, we recover the Kan operation for cubical sets [53].

6.4.5 Equality Judgments for Composition

The equality judgments for $\mathsf{comp}^i\ C\ [\varphi \mapsto u]\ a_0$ are defined by cases on the type C which depends on i, i.e., $\Gamma, i : \mathbb{I} \vdash C$. The right hand side of the definitions are all equal to $u(i1)$ on the extent φ by the typing rule for compositions. There are four cases to consider:

Product Types, $C = (x : A) \to B$

Given $\Gamma, \varphi, i : \mathbb{I} \vdash \mu : C$ and $\Gamma \vdash \lambda_0 : C(i0)[\varphi \mapsto \mu(i0)]$ the composition will be of type $C(i1)$. For $\Gamma \vdash u_1 : A(i1)$, we first let:

$$\begin{array}{ll} w = \mathsf{fill}^i\ A(i/1 - i)\ []\ u_1 & \text{(in context } \Gamma, i : \mathbb{I} \text{ and of type } A(i/1 - i)) \\ v = w(i/1 - i) & \text{(in context } \Gamma, i : \mathbb{I} \text{ and of type } A) \end{array}$$

Using this we define the equality judgment:

$$\Gamma \vdash (\mathsf{comp}^i\ C\ [\varphi \mapsto \mu]\ \lambda_0)\ u_1 = \\ \mathsf{comp}^i\ B(x/v)\ [\varphi \mapsto \mu\ v]\ (\lambda_0\ v(i0)) : B(x/v)(i1)$$

Sum Types, $C = (x : A) \times B$

Given $\Gamma, \varphi, i : \mathbb{I} \vdash w : C$ and $\Gamma \vdash w_0 : C(i0)[\varphi \mapsto w(i0)]$ we let:

$$\begin{array}{ll} a = \mathsf{fill}^i\ A\ [\varphi \mapsto w.1]\ w_0.1 & \text{(in context } \Gamma, i : \mathbb{I} \text{ and of type } A) \\ c_1 = \mathsf{comp}^i\ A\ [\varphi \mapsto w.1]\ w_0.1 & \text{(in context } \Gamma \text{ and of type } A(i1)) \\ c_2 = \mathsf{comp}^i\ B(x/a)\ [\varphi \mapsto w.2]\ w_0.2 & \text{(in context } \Gamma \text{ and of type } B(x/a)(i1)) \end{array}$$

From which we define:

$$\Gamma \vdash \mathsf{comp}^i\ C\ [\varphi \mapsto w]\ w_0 = (c_1, c_2) : C(i1)$$

Natural Numbers, $C = \mathsf{N}$

In this we define $\mathsf{comp}^i\ C\ [\varphi \mapsto n]\ n_0$ by recursion:

$$\Gamma \vdash \mathsf{comp}^i\ C\ [\varphi \mapsto \mathsf{0}]\ \mathsf{0} = \mathsf{0} : C$$
$$\Gamma \vdash \mathsf{comp}^i\ C\ [\varphi \mapsto \mathsf{s}\ n]\ (\mathsf{s}\ n_0) = \mathsf{s}\ (\mathsf{comp}^i\ C\ [\varphi \mapsto n]\ n_0) : C$$

Path Types, $C = \mathsf{Path}\ A\ u\ v$

Given $\Gamma, \varphi, i : \mathbb{I} \vdash p : C$ and $\Gamma \vdash p_0 : C(i0)[\varphi \mapsto p(i0)]$ we define:

$$\Gamma \vdash \mathsf{comp}^i\ C\ [\varphi \mapsto p]\ p_0 = \langle j \rangle\ \mathsf{comp}^i\ A\ [\varphi \mapsto p\ j, (j = 0) \mapsto u, (j = 1) \mapsto v]\ (p_0\ j) : C(i1)$$

6.4.6 Transport

Composition for $\varphi = 0_{\mathbb{F}}$ corresponds to transport:

$$\Gamma \vdash \mathsf{transp}^i\ A\ a = \mathsf{comp}^i\ A\ []\ a : A(i1)$$

Together with the fact that singletons are contractible, from Section 6.3.2, we get the elimination principle for Path types in the same manner as explained for identity types in Section 7.2 of [15].

6.5 Derived Notions and Operations

This section defines various notions and operations that will be used for defining compositions for the glue operation in the next section. This operation will then be used to define the composition operation for the universe and to prove the univalence axiom.

6.5.1 Contractible Types

We define $\mathsf{isContr}\ A = (x : A) \times ((y : A) \to \mathsf{Path}\ A\ x\ y)$. A proof of $\mathsf{isContr}\ A$ witnesses the fact that A is *contractible*.

Given $\Gamma \vdash p : \mathsf{isContr}\ A$ and $\Gamma, \varphi \vdash u : A$ we define the operation[3]

$$\Gamma \vdash \mathsf{contr}\ p\ [\varphi \mapsto u] = \mathsf{comp}^i\ A\ [\varphi \mapsto p.2\ u\ i]\ p.1 : A[\varphi \mapsto u].$$

Conversely, we have the following characterization of contractible types:

Lemma 6.5.1. *Let* $\Gamma \vdash A$ *and assume that we have one operation*

$$\frac{\Gamma, \varphi \vdash u : A}{\Gamma \vdash \mathsf{contr}\ [\varphi \mapsto u] : A[\varphi \mapsto u]}$$

then we can find an element in $\mathsf{isContr}\ A$.

[3] This expresses that the restriction map $\Gamma, \varphi \to \Gamma$ has the left lifting property w.r.t. any "trivial fibration", i.e., contractible extensions $\Gamma, x : A \to \Gamma$. The restriction maps $\Gamma, \varphi \to \Gamma$ thus represent "cofibrations" while the maps $\Gamma, x : A \to \Gamma$ represent "fibrations".

Proof. We define $x = \mathsf{contr}\ [] : A$ and prove that any element $y : A$ is path equal to x. For this, we introduce a fresh name $i : \mathbb{I}$ and define $\varphi = (i = 0) \vee (i = 1)$ and $u = [(i = 0) \mapsto x, (i = 1) \mapsto y]$. Using this we obtain $\Gamma, i : \mathbb{I} \vdash v = \mathsf{contr}\ [\varphi \mapsto u] : A[\varphi \mapsto u]$. In this way, we get a path $\langle i \rangle\ \mathsf{contr}\ [\varphi \mapsto u]$ connecting x and y. □

6.5.2 The pres Operation

The pres operation states that the image of a composition is path equal to the composition of the respective images, so that any function *preserves* composition, up to path equality.

Lemma 6.5.2. *We have an operation:*

$$\frac{\begin{array}{c}\Gamma, i : \mathbb{I} \vdash f : T \to A \\ \Gamma \vdash \varphi : \mathbb{F} \qquad \Gamma, \varphi, i : \mathbb{I} \vdash t : T \qquad \Gamma \vdash t_0 : T(i0)[\varphi \mapsto t(i0)]\end{array}}{\Gamma \vdash \mathsf{pres}^i\ f\ [\varphi \mapsto t]\ t_0 : (\mathsf{Path}\ A(i1)\ c_1\ c_2)[\varphi \mapsto \langle j \rangle\ (f\ t)(i1)]}$$

where $c_1 = \mathsf{comp}^i\ A\ [\varphi \mapsto f\ t]\ (f(i0)\ t_0)$ *and* $c_2 = f(i1)\ (\mathsf{comp}^i\ T\ [\varphi \mapsto t]\ t_0)$.

Proof. Let $\Gamma \vdash a_0 = f(i0)\ t_0 : A(i0)$ and $\Gamma, i : \mathbb{I} \vdash v = \mathsf{fill}^i\ T\ [\varphi \mapsto t]\ t_0 : T$. We take $\mathsf{pres}^i\ f\ [\varphi \mapsto t]\ t_0 = \langle j \rangle\ \mathsf{comp}^i\ A\ [\varphi \vee (j = 1) \mapsto f\ v]\ a_0$. □

Note that pres^i binds i in f and t.

6.5.3 The equiv Operation

We define $\mathsf{isEquiv}\ T\ A\ f = (y : A) \to \mathsf{isContr}\ ((x : T) \times \mathsf{Path}\ A\ y\ (f\ x))$ and $\mathsf{Equiv}\ T\ A = (f : T \to A) \times \mathsf{isEquiv}\ T\ A\ f$. If $f : \mathsf{Equiv}\ T\ A$ and $t : T$, we may write $f\ t$ for $f.1\ t$.

Lemma 6.5.3. *If* $\Gamma \vdash f : \mathsf{Equiv}\ T\ A$*, we have an operation:*

$$\frac{\Gamma, \varphi \vdash t : T \qquad \Gamma \vdash a : A \qquad \Gamma, \varphi \vdash p : \mathsf{Path}\ A\ a\ (f\ t)}{\Gamma \vdash \mathsf{equiv}\ f\ [\varphi \mapsto (t, p)]\ a : ((x : T) \times \mathsf{Path}\ A\ a\ (f\ x))[\varphi \mapsto (t, p)]}$$

Conversely, if $\Gamma \vdash f : T \to A$ *and we have such an operation, then we can build a proof that* f *is an equivalence.*

Proof. We define $\mathsf{equiv}\ f\ [\varphi \mapsto (t, p)]\ a = \mathsf{contr}\ (f.2\ a)\ [\varphi \mapsto (t, p)]$ using the contr operation defined above. The second statement follows from Lemma 6.5.1. □

6.6 Glueing

In this section, we introduce the glueing operation. This operation expresses that to be "extensible" is invariant by equivalence. From this operation, we can define a composition operation for universes, and prove the univalence axiom.

6.6.1 Syntax and Inference Rules for Glueing

We introduce the *glueing* construction at type and term level by:

$$
\begin{array}{lcll}
t, u, A, B & ::= & \ldots & \\
& | & \mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A & \text{Glue type} \\
& | & \mathsf{glue}\ [\varphi \mapsto t]\ u & \text{Glue term} \\
& | & \mathsf{unglue}\ [\varphi \mapsto f]\ u & \text{Unglue term}
\end{array}
$$

We may write simply $\mathsf{unglue}\ b$ for $\mathsf{unglue}\ [\varphi \mapsto f]\ b$. The inference rules for these are presented in Figure 6.5.

$$\frac{\Gamma \vdash A \qquad \Gamma, \varphi \vdash T \qquad \Gamma, \varphi \vdash f : \mathsf{Equiv}\ T\ A}{\Gamma \vdash \mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A}$$

$$\frac{\Gamma \vdash b : \mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A}{\Gamma \vdash \mathsf{unglue}\ b : A[\varphi \mapsto f\ b]}$$

$$\frac{\Gamma, \varphi \vdash f : \mathsf{Equiv}\ T\ A \qquad \Gamma, \varphi \vdash t : T \qquad \Gamma \vdash a : A[\varphi \mapsto f\ t]}{\Gamma \vdash \mathsf{glue}\ [\varphi \mapsto t]\ a : \mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A}$$

$$\frac{\Gamma \vdash T \qquad \Gamma \vdash f : \mathsf{Equiv}\ T\ A}{\Gamma \vdash \mathsf{Glue}\ [1_{\mathbb{F}} \mapsto (T, f)]\ A = T} \qquad \frac{\Gamma \vdash t : T \qquad \Gamma \vdash f : \mathsf{Equiv}\ T\ A}{\Gamma \vdash \mathsf{glue}\ [1_{\mathbb{F}} \mapsto t]\ (f\ t) = t : T}$$

$$\frac{\Gamma \vdash b : \mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A}{\Gamma \vdash b = \mathsf{glue}\ [\varphi \mapsto b]\ (\mathsf{unglue}\ b) : \mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A}$$

$$\frac{\Gamma, \varphi \vdash f : \mathsf{Equiv}\ T\ A \qquad \Gamma, \varphi \vdash t : T \qquad \Gamma \vdash a : A[\varphi \mapsto f\ t]}{\Gamma \vdash \mathsf{unglue}\ (\mathsf{glue}\ [\varphi \mapsto t]\ a) = a : A}$$

Figure 6.5: Inference rules for glueing

It follows from these rules that if $\Gamma \vdash b : \mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A$, then $\Gamma, \varphi \vdash b : T$.

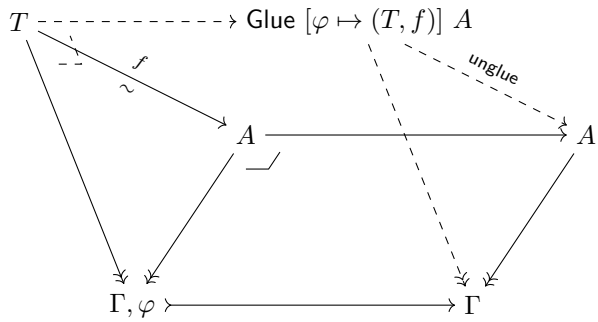
In the case $\varphi = (i = 0) \vee (i = 1)$ the glueing operation can be illustrated as the dashed line in:

$$
\begin{array}{ccc}
T_0 & \dashrightarrow & T_1 \\
{\scriptstyle f(i0)}\downarrow\wr & & \wr\downarrow{\scriptstyle f(i1)} \\
A(i0) & \xrightarrow[A]{} & A(i1)
\end{array}
$$

This illustrates why the operation is called glue: it *glues* together along a

partial equivalence the partial type T and the total type A to a total type that extends T.

Remark 6.6.1. In general $\mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A$ can be illustrated as:



This diagram suggests that a construction similar to Glue also appears in the simplicial set model. Indeed, the proof of Theorem 3.4.1 in [54] contains a similar diagram where $\overline{E}_1$ corresponds to $\mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A$.

Example 6.6.2. Using glueing we can construct a path from an equivalence $\Gamma \vdash f : \mathsf{Equiv}\ A\ B$ by defining

$$\Gamma, i : \mathbb{I} \vdash E = \mathsf{Glue}\ [(i = 0) \mapsto (A, f), (i = 1) \mapsto (B, \mathsf{id}_B)]\ B$$

so that $E(i0) = A$ and $E(i1) = B$, where $\mathsf{id}_B : \mathsf{Equiv}\ B\ B$ is defined as:

$$(\lambda x : B.\ x, \lambda x : B.\ ((x, 1_x), \lambda u : (y : B) \times \mathsf{Path}\ B\ x\ y.\ \langle i \rangle\ (u.2\ i, \langle j \rangle\ u.2\ (i \wedge j))))$$

In Section 6.7 we introduce a universe of types U and we will be able to define a function of type $(A\ B : \mathsf{U}) \to \mathsf{Equiv}\ A\ B \to \mathsf{Path}\ \mathsf{U}\ A\ B$ by:

$$\lambda A\ B : \mathsf{U}.\ \lambda f : \mathsf{Equiv}\ A\ B.\ \langle i \rangle\ \mathsf{Glue}\ [(i = 0) \mapsto (A, f), (i = 1) \mapsto (B, \mathsf{id}_B)]\ B$$

6.6.2 Composition for Glueing

We assume $\Gamma, i : \mathbb{I} \vdash B = \mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A$, and define the composition in B. In order to do so, assume

$$\Gamma, \psi, i : \mathbb{I} \vdash b : B \qquad\qquad \Gamma \vdash b_0 : B(i0)[\psi \mapsto b(i0)]$$

and define:

$$\begin{aligned} \Gamma, \psi, i : \mathbb{I} \vdash a &:= \mathsf{unglue}\ b : A[\varphi \mapsto f\ b] \\ \Gamma \vdash a_0 &:= \mathsf{unglue}\ b_0 : A(i0)[\varphi(i0) \mapsto f(i0)\ b_0, \psi \mapsto a(i0)] \end{aligned}$$

The following provides the algorithm for composition $\mathsf{comp}^i\ B\ [\psi \mapsto b]\ b_0 = b_1$ of type $B(i1)[\psi \mapsto b(i1)]$.

$$\begin{array}{lcll}
\delta & = & \forall i.\varphi & \Gamma \\
a'_1 & = & \mathsf{comp}^i\ A\ [\psi \mapsto a]\ a_0 & \Gamma \\
t'_1 & = & \mathsf{comp}^i\ T\ [\psi \mapsto b]\ b_0 & \Gamma, \delta \\
\omega & = & \mathsf{pres}^i\ f\ [\psi \mapsto b]\ b_0 & \Gamma, \delta \\
(t_1, \alpha) & = & \mathsf{equiv}\ f(i1)\ [\delta \mapsto (t'_1, \omega), \psi \mapsto (b(i1), \langle j \rangle a'_1)]\ a'_1 & \Gamma, \varphi(i1) \\
a_1 & = & \mathsf{comp}^j\ A(i1)\ [\varphi(i1) \mapsto \alpha\ j, \psi \mapsto a(i1)]\ a'_1 & \Gamma \\
b_1 & = & \mathsf{glue}\ [\varphi(i1) \mapsto t_1]\ a_1 & \Gamma
\end{array}$$

We can check that whenever $\Gamma, i : \mathbb{I} \vdash \varphi = 1_{\mathbb{F}} : \mathbb{F}$ the definition of b_1 coincides with $\mathsf{comp}^i\ T\ [\psi \mapsto b]\ b_0$, which is consistent with the fact that $B = T$ in this case.

In the next section we will use the glue operation to define the composition for the universe and to prove the univalence axiom.

6.7 Universe and the Univalence Axiom

As in [62], we now introduce a universe U à la Russell by reflecting all typing rules and:

$$\frac{\Gamma \vdash}{\Gamma \vdash \mathsf{U}} \qquad\qquad \frac{\Gamma \vdash A : \mathsf{U}}{\Gamma \vdash A}$$

In particular, we have $\Gamma \vdash \mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A : \mathsf{U}$ whenever $\Gamma \vdash A : \mathsf{U}$, $\Gamma, \varphi \vdash T : \mathsf{U}$, and $\Gamma, \varphi \vdash f : \mathsf{Equiv}\ T\ A$.

6.7.1 Composition for the Universe

In order to describe the composition operation for the universe we first have to explain how to construct an equivalence from a line in the universe. Given $\Gamma \vdash A$, $\Gamma \vdash B$, and $\Gamma, i : \mathbb{I} \vdash E$, such that $E(i0) = A$ and $E(i1) = B$, we will construct $\mathsf{equiv}^i\ E : \mathsf{Equiv}\ A\ B$. In order to do this we first define

$$\begin{array}{rl}
\Gamma \vdash f := & \lambda x : A.\ \mathsf{transp}^i\ E\ x : A \to B \\
\Gamma \vdash g := & \lambda y : B.\ (\mathsf{transp}^i\ E(i/1-i)\ y)(i/1-i) : B \to A \\
\Gamma, i : \mathbb{I} \vdash u := & \lambda x : A.\ \mathsf{fill}^i\ E\ []\ x : A \to E \\
\Gamma, i : \mathbb{I} \vdash v := & \lambda y : B.\ (\mathsf{fill}^i\ E(i/1-i)\ []\ y)(i/1-i) : B \to E
\end{array}$$

such that:

$$u(i0) = \lambda x : A.x \qquad u(i1) = f \qquad v(i0) = g \qquad v(i1) = \lambda y : B.y$$

We will now prove that f is an equivalence. Given $y : B$ we see that $(x : A) \times \mathsf{Path}\ B\ y\ (f\ x)$ is inhabited as it contains the element $(g\ y, \langle j \rangle\ \theta_0(i1))$ where

$$\theta_0 = \mathsf{fill}^i\ E\ [(j = 0) \mapsto v\ y, (j = 1) \mapsto u\ (g\ y)]\ (g\ y).$$

Next, given an element (x, β) of $(x : A) \times \mathsf{Path}\ B\ y\ (f\ x)$ we will construct a path from $(g\ y, \langle j \rangle\ \theta_0(i1))$ to (x, β). Let

$$\theta_1 = \big(\mathsf{fill}^i\ E(i/1 - i) \\ [(j = 0) \mapsto (v\ y)(i/1 - i), (j = 1) \mapsto (u\ x)(i/1 - i)]\ (\beta\ j)\big)(i/1 - i)$$

and $\omega = \theta_1(i0)$ so $\Gamma, i : \mathbb{I}, j : \mathbb{I} \vdash \theta_1 : E$, $\omega(j0) = g\ y$, and $\omega(j1) = x$. And further with

$$\delta = \mathsf{comp}^i\ E\ [(k = 0) \mapsto \theta_0, (k = 1) \mapsto \theta_1, \\ (j = 0) \mapsto v\ y, (j = 1) \mapsto u\ \omega(j/k)]\ \omega(j/j \wedge k)$$

we obtain

$$\langle k \rangle\ (\omega(j/k), \langle j \rangle\ \delta) : \mathsf{Path}\ ((x : A) \times \mathsf{Path}\ B\ y\ (f\ x))\ (g\ y, \langle j \rangle\ \theta_0(i1))\ (x, \beta)$$

as desired. This concludes the proof that f is an equivalence and thus also the construction of $\mathsf{equiv}^i\ E : \mathsf{Equiv}\ A\ B$.

Using this we can now define the composition for the universe:

$$\Gamma \vdash \mathsf{comp}^i\ \mathsf{U}\ [\varphi \mapsto E]\ A_0 = \mathsf{Glue}\ [\varphi \mapsto (E(i1), \mathsf{equiv}^i\ E(i/1 - i))]\ A_0 : \mathsf{U}$$

Remark 6.7.1. Given $\Gamma, i : \mathbb{I} \vdash E$ we can also get an equivalence in $\mathsf{Equiv}\ A\ B$ (where $A = E(i0)$ and $B = E(i1)$) with a less direct description by

$$\Gamma \vdash \mathsf{transp}^i\ (\mathsf{Equiv}\ A\ E)\ \mathsf{id}_A : \mathsf{Equiv}\ A\ B$$

where id_A is the identity equivalence as given in Example 6.6.2.

6.7.2 The Univalence Axiom

Given $B = \mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A$ the map $\mathsf{unglue} : B \to A$ extends f, in the sense that $\Gamma, \varphi \vdash \mathsf{unglue}\ b = f\ b : A$ if $\Gamma \vdash b : B$.

Theorem 6.7.2. *The map* $\mathsf{unglue} : B \to A$ *is an equivalence.*

Proof. By Lemma 6.5.3 it suffices to construct

$$\tilde{b} : B[\psi \mapsto b] \qquad\qquad \tilde{\alpha} : \mathsf{Path}\ A\ u\ (\mathsf{unglue}\ \tilde{b})[\psi \mapsto \alpha]$$

given $\Gamma, \psi \vdash b : B$ and $\Gamma \vdash u : A$ and $\Gamma, \psi \vdash \alpha : \mathsf{Path}\ A\ u\ (\mathsf{unglue}\ b)$.

Since $\Gamma, \varphi \vdash f : T \to A$ is an equivalence and

$$\Gamma, \varphi, \psi \vdash b : T \qquad\qquad \Gamma, \varphi, \psi \vdash \alpha : \mathsf{Path}\ A\ u\ (f\ b)$$

we get, using Lemma 6.5.3:

$$\Gamma, \varphi \vdash t : T[\psi \mapsto b] \qquad\qquad \Gamma, \varphi \vdash \beta : \mathsf{Path}\ A\ u\ (f\ t)\ [\psi \mapsto \alpha]$$

We then define $\tilde{a} = \mathsf{comp}^i\ A\ [\varphi \mapsto \beta\ i, \psi \mapsto \alpha\ i]\ u$, and using this we conclude by letting $\tilde{b} = \mathsf{glue}\ [\varphi \mapsto t]\ \tilde{a}$ and $\tilde{\alpha} = \mathsf{fill}^i\ A\ [\varphi \mapsto \beta\ i, \psi \mapsto \alpha\ i]\ u$. □

Corollary 6.7.3. *For any type* $A : \mathsf{U}$ *the type* $C = (X : \mathsf{U}) \times \mathsf{Equiv}\ X\ A$ *is contractible.*[4]

Proof. It is enough by Lemma 6.5.1 to show that any partial element $\varphi \vdash (T, f) : C$ is path equal to the restriction of a total element. The map unglue extends f and is an equivalence by the previous theorem. Since any two elements of the type $\mathsf{isEquiv}\ X\ A\ f.1$ are path equal, this shows that any partial element of type C is path equal to the restriction of a total element. We can then conclude by Theorem 6.7.2. □

Corollary 6.7.4 (Univalence axiom)**.** *For any term*

$$t : (A\ B : \mathsf{U}) \to \mathsf{Path}\ \mathsf{U}\ A\ B \to \mathsf{Equiv}\ A\ B$$

the map $t\ A\ B : \mathsf{Path}\ \mathsf{U}\ A\ B \to \mathsf{Equiv}\ A\ B$ *is an equivalence.*

Proof. Both $(X : \mathsf{U}) \times \mathsf{Path}\ \mathsf{U}\ A\ X$ and $(X : \mathsf{U}) \times \mathsf{Equiv}\ A\ X$ are contractible. Hence the result follows from Theorem 4.7.7 in [79]. □

Two alternative proofs of univalence can be found in Appendix 6.B.

6.8 Semantics

In this section we will explain the semantics of the type theory under consideration in cubical sets. We will first review how cubical sets, as a presheaf category, yield a model of basic type theory, and then explain the additional so-called composition structure we have to require to interpret the full cubical type theory.

6.8.1 The Category of Cubes and Cubical Sets

Consider the monad dM on the category of sets associating to each set the free de Morgan algebra on that set. The *category of cubes* $\mathcal{C}$ is the small category whose objects are finite subsets $I, J, K, \ldots$ of a fixed, discrete, and countably infinite set, called *names*, and a morphism $\mathrm{Hom}(J, I)$ is a map $I \to \mathsf{dM}(J)$. Identities and compositions are inherited from the Kleisli category of dM, i.e., the identity on I is given by the unit $I \to \mathsf{dM}(I)$, and composition $fg \in \mathrm{Hom}(K, I)$ of $g \in \mathrm{Hom}(K, J)$ and $f \in \mathrm{Hom}(J, I)$ is given by $\mu_K \circ \mathsf{dM}(g) \circ f$ where $\mu_K \colon \mathsf{dM}(\mathsf{dM}(K)) \to \mathsf{dM}(K)$ denotes multiplication of dM. We will use f, g, h for morphisms in $\mathcal{C}$ and simply write $f \colon J \to I$ for $f \in \mathrm{Hom}(J, I)$. We

[4] This formulation of the univalence axiom can be found in the message of Martín Escardó in:
`https://groups.google.com/forum/#!msg/homotopytypetheory/HfCB_b-PNEU/Ibb48LvUMeUJ`
This is also used in the (classical) proofs of the univalence axiom, see Theorem 3.4.1 of [54] and Proposition 2.18 of [24], where an operation similar to the glueing operation appears implicitly.

will often write unions with commas and omit curly braces around finite sets of names, e.g., writing I, i, j for $I \cup \{i, j\}$ and $I - i$ for $I - \{i\}$ etc.

If i is in I and b is $0_{\mathbb{I}}$ or $1_{\mathbb{I}}$, we have maps (ib) in $\mathrm{Hom}(I - i, I)$ whose underlying map sends $j \neq i$ to itself and i to b. A *face map* is a composition of such maps. A *strict map* $\mathrm{Hom}(J, I)$ is a map $I \to \mathsf{dM}(J)$ which never takes the value $0_{\mathbb{I}}$ or $1_{\mathbb{I}}$. Any f can be uniquely written as a composition $f = gh$ where g is a face map and h is strict.

Definition 6.8.1. A *cubical set* is a presheaf on $\mathcal{C}$.

Thus, a cubical set Γ is given by sets $\Gamma(I)$ for each $I \in \mathcal{C}$ and maps (called restrictions) $\Gamma(f)\colon \Gamma(I) \to \Gamma(J)$ for each $f\colon J \to I$. If we write $\Gamma(f)(\rho) = \rho f$ for $\rho \in \Gamma(I)$ (leaving the Γ implicit), these maps should satisfy $\rho\,\mathrm{id}_I = \rho$ and $(\rho f)g = \rho(fg)$ for $f\colon J \to I$ and $g\colon K \to J$.

Let us discuss some important examples of cubical sets. Using the canonical de Morgan algebra structure of the unit interval, $[0, 1]$, we can define a functor

$$\mathcal{C} \to \mathbf{Top}, \quad I \mapsto [0, 1]^I. \tag{6.1}$$

If u is in $[0, 1]^I$ we can think of u as an environment giving values in $[0, 1]$ to each $i \in I$, so that iu is in $[0, 1]$ if $i \in I$. Since $[0, 1]$ is a de Morgan algebra, this extends uniquely to ru for $r \in \mathsf{dM}(I)$. So any $f\colon J \to I$ in $\mathcal{C}$ induces $f\colon [0, 1]^J \to [0, 1]^I$ by $i(fu) = (if)u$.

To any topological space X we can associate its *singular cubical set* $\mathrm{S}(X)$ by taking $\mathrm{S}(X)(I)$ to be the set of continuous functions $[0, 1]^I \to X$.

For a finite set of names I we get the formal cube $\mathbf{y}\,I$ where $\mathbf{y}\colon \mathcal{C} \to [\mathcal{C}^{\mathrm{op}}, \mathbf{Set}]$ denotes the Yoneda embedding. Note that since **Top** is cocomplete the functor in (6.1) extends to a cocontinuous functor assigning to each cubical set its *geometric realization* as a topological space, in such a way that $\mathbf{y}\,I$ has $[0, 1]^I$ as its geometric realization.

The formal interval $\mathbb{I}$ induces a cubical set given by $\mathbb{I}(I) = \mathsf{dM}(I)$. The face lattice $\mathbb{F}$ induces a cubical set by taking as $\mathbb{F}(I)$ to be those $\varphi \in \mathbb{F}$ which only use symbols in I. The restrictions along $f\colon J \to I$ are in both cases simply *substituting* the symbols $i \in I$ by $f(i) \in \mathsf{dM}(J)$.

As any presheaf category, cubical sets have a subobject classifier Ω where $\Omega(I)$ is the set of sieves on I (i.e., subfunctors of $\mathbf{y}\,I$). Consider the natural transformation $(\cdot = 1)\colon \mathbb{I} \to \Omega$ where for $r \in \mathbb{I}(I)$, $(r = 1)$ is the sieve on I of all $f\colon J \to I$ such that $rf = 1_{\mathbb{I}}$. The image of $(\cdot = 1)$ is $\mathbb{F} \to \Omega$, assigning to each φ the sieve of all f with $\varphi f = 1_{\mathbb{F}}$.

6.8.2 Presheaf Semantics

The category of cubical sets (with morphisms being natural transformations) induce—as does any presheaf category—a category with families (CwF) [34] where the category of contexts and substitutions is the category of cubical sets. We will review the basic constructions but omit verification of the required equations (see, e.g., [47, 50, 15] for more details).

Basic Presheaf Semantics

As already mentioned the category of (semantic) contexts and substitutions is given by cubical sets and their maps. In this section we will use Γ, Δ to denote cubical sets and (semantic) substitutions by $\sigma \colon \Delta \to \Gamma$, overloading previous use of the corresponding meta-variables to emphasize their intended role.

Given a cubical set Γ, the types A in context Γ, written $A \in \mathrm{Ty}(\Gamma)$, are given by sets $A\rho$ for each $I \in \mathcal{C}$ and $\rho \in \Gamma(I)$ together with restriction maps $A\rho \to A(\rho f)$, $u \mapsto uf$ for $f \colon J \to I$ satisfying $u\,\mathrm{id}_I = u$ and $(uf)g = u(fg) \in A(\rho fg)$ if $g \colon K \to J$. Equivalently, $A \in \mathrm{Ty}(\Gamma)$ are the presheaves on the category of elements of Γ. For a type $A \in \mathrm{Ty}(\Gamma)$ its terms $a \in \mathrm{Ter}(\Gamma; A)$ are given by families of elements $a\rho \in A\rho$ for each $I \in \mathcal{C}$ and $\rho \in \Gamma(I)$ such that $(a\rho)f = a(\rho f)$ for $f \colon J \to I$. Note that our notation leaves a lot implicit; e.g., we should have written $A(I, \rho)$ for $A\rho$; $A(I, \rho, f)$ for the restriction map $A\rho \to A(\rho f)$; and $a(I, \rho)$ for $a\rho$.

For $A \in \mathrm{Ty}(\Gamma)$ and $\sigma \colon \Delta \to \Gamma$ we define $A\sigma \in \mathrm{Ty}(\Delta)$ by $(A\sigma)\rho = A(\sigma\rho)$ and the induced restrictions. If we also have $a \in \mathrm{Ter}(\Gamma; A)$, we define $a\sigma \in \mathrm{Ter}(\Delta; A\sigma)$ by $(a\sigma)\rho = a(\sigma\rho)$. For a type $A \in \mathrm{Ty}(\Gamma)$ we define the cubical set $\Gamma.A$ by $(\Gamma.A)(I)$ being the set of all (ρ, u) with $\rho \in \Gamma(I)$ and $u \in A\rho$; restrictions are given by $(\rho, u)f = (\rho f, uf)$. The first projection yields a map $\mathsf{p} \colon \Gamma.A \to \Gamma$ and the second projection a term $\mathsf{q} \in \mathrm{Ter}(\Gamma.A; A\mathsf{p})$. Given $\sigma \colon \Delta \to \Gamma$, $A \in \mathrm{Ty}(\Gamma)$, and $a \in \mathrm{Ter}(\Delta; A\sigma)$ we define $(\sigma, a) \colon \Delta \to \Gamma.A$ by $(\sigma, a)\rho = (\sigma\rho, a\rho)$. For $u \in \mathrm{Ter}(\Gamma; A)$ we define $[u] = (\mathrm{id}_\Gamma, u) \colon \Gamma \to \Gamma.A$.

The basic type formers are interpreted as follows. For $A \in \mathrm{Ty}(\Gamma)$ and $B \in \mathrm{Ty}(\Gamma.A)$ define $\Sigma_\Gamma(A, B) \in \mathrm{Ty}(\Gamma)$ by letting $\Sigma_\Gamma(A, B)\rho$ contain all pairs (u, v) where $u \in A\rho$ and $v \in B(\rho, v)$; restrictions are defined as $(u, v)f = (uf, vf)$. Given $w \in \mathrm{Ter}(\Gamma; \Sigma(A, B))$ we get $w.1 \in \mathrm{Ter}(\Gamma; A)$ and $w.2 \in \mathrm{Ter}(\Gamma; B[w.1])$ by $(w.1)\rho = \mathsf{p}(w\rho)$ and $(w.2)\rho = \mathsf{q}(w\rho)$ where $\mathsf{p}(u, v) = u$ and $\mathsf{q}(u, v) = v$ are the set-theoretic projections.

Given $A \in \mathrm{Ty}(\Gamma)$ and $B \in \mathrm{Ty}(\Gamma.A)$ the dependent function space $\Pi_\Gamma(A, B)$ in $\mathrm{Ty}(\Gamma)$ is defined by letting $\Pi_\Gamma(A, B)\rho$ for $\rho \in \Gamma(I)$ contain all families $w = (w_f \mid J \in \mathcal{C}, f \colon J \to I)$ where

$$w_f \in \prod_{u \in A(\rho f)} B(\rho f, u) \text{ such that } (w_f\, u)g = w_{fg}(ug) \text{ for } u \in A(\rho f),\ g \colon K \to J.$$

The restriction by $f \colon J \to I$ of such a w is defined by $(wf)_g = w_{fg}$. Given $v \in \mathrm{Ter}(\Gamma.A; B)$ we have $\lambda_{\Gamma;A} v \in \mathrm{Ter}(\Gamma; \Pi(A, B))$ given by $((\lambda v)\rho)_f\, u = v(\rho f, u)$. Application $\mathsf{app}(w, u) \in \mathrm{Ter}(\Gamma; B[u])$ of $w \in \mathrm{Ter}(\Gamma; \Pi(A, B))$ to $u \in \mathrm{Ter}(\Gamma; A)$ is defined by

$$\mathsf{app}(w, u)\rho = (w\rho)_{\mathrm{id}_I}(u\rho) \in (B[u])\rho. \tag{6.2}$$

Basic data types like the natural numbers can be interpreted as discrete presheaves, i.e., $\mathtt{N} \in \mathrm{Ty}(\Gamma)$ is given by $\mathtt{N}\rho = \mathbb{N}$; the constants are interpreted by the lifts of the corresponding set-theoretic operations on $\mathbb{N}$. This concludes the outline of the basic CwF structure on cubical sets.

Remark 6.8.2. Following Aczel [3] we will make use of that our semantic entities are actual sets in the ambient set theory. This will allow us to interpret syntax in Section 6.8.3 with fewer type annotations than are usually needed for general categorical semantics of type theory (see [75]). E.g., the definition of application $\mathtt{app}(w,u)\rho$ as defined in (6.2) is independent of Γ, A and B, since set-theoretic application is a (class) operation on all sets. Likewise, we don't need annotations for first and second projections. But note that we will need the type A for λ-abstraction for $(\lambda_{\Gamma;A} v)\rho$ to be a set by the replacement axiom.

Semantic Path Types

Note that we can consider any cubical set X as $X' \in \mathrm{Ty}(\Gamma)$ by setting $X'\rho = X(I)$ for $\rho \in \Gamma(I)$. We will usually simply write X for X'. In particular, for a cubical set Γ we can form the cubical set $\Gamma.\mathbb{I}$.

For $A \in \mathrm{Ty}(\Gamma)$ and $u, v \in \mathrm{Ter}(\Gamma; A)$ the semantic path type $\mathtt{Path}^{\Gamma}_{A}(u,v) \in \mathrm{Ty}(\Gamma)$ is given by: for $\rho \in \Gamma(I)$, $\mathtt{Path}_A(u,v)\rho$ consists of equivalence classes $\langle i\rangle\ w$ where $i \notin I$, $w \in A(\rho \mathrm{s}_i)$ such that $w(i0) = u\rho$ and $w(i1) = v\rho$; two such elements $\langle i\rangle\ w$ and $\langle j\rangle\ w'$ are equal iff $w(i/j) = w'$. Here $\mathrm{s}_i \colon I, i \to I$ is induced by the inclusion $I \subseteq I, i$ and (i/j) setting i to j. We define $(\langle i\rangle\ w)f = \langle j\rangle\ w(f, i/j)$ for $f \colon J \to I$ and $j \notin J$. For $r \in \mathbb{I}(I)$ we set $(\langle i\rangle\ w)\, r = w(i/r)$. Both operations, name abstraction and application, lift to terms, i.e., if $w \in \mathrm{Ter}(\Gamma.\mathbb{I}; A)$, then $\langle\rangle w \in \mathrm{Ter}(\Gamma; \mathtt{Path}_A(w[0], w[1]))$ given by $(\langle\rangle w)\rho = \langle i\rangle\ w(\rho \mathrm{s}_i)$ for a fresh i; also if $u \in \mathrm{Ter}(\Gamma; \mathtt{Path}_A(a,b))$ and $r \in \mathrm{Ter}(\Gamma; \mathbb{I})$, then $u\ r \in \mathrm{Ter}(\Gamma; A)$ defined as $(u\ r)\rho = (u\rho)\ (r\rho)$.

Composition Structure

For $\varphi \in \mathrm{Ter}(\Gamma; \mathbb{F})$ we define the cubical set Γ, φ by taking $\rho \in (\Gamma, \varphi)(I)$ iff $\rho \in \Gamma(I)$ and $\varphi\rho = 1_{\mathbb{F}} \in \mathbb{F}$; the restrictions are those induced by Γ. In particular, we have $\Gamma, 1 = \Gamma$ and $\Gamma, 0$ is the empty cubical set. (Here, $0 \in \mathrm{Ter}(\Gamma; \mathbb{F})$ is $0\rho = 0_{\mathbb{F}}$ and similarly for $1_{\mathbb{F}}$.) Any $\sigma \colon \Delta \to \Gamma$ gives rise to a morphism $\Delta, \varphi\sigma \to \Gamma, \varphi$ which we also will denote by σ.

If $A \in \mathrm{Ty}(\Gamma)$ and $\varphi \in \mathrm{Ter}(\Gamma; \mathbb{F})$, we define a *partial element of* $A \in \mathrm{Ty}(\Gamma)$ *of extent* φ to be an element of $\mathrm{Ter}(\Gamma, \varphi; A\iota_\varphi)$ where $\iota_\varphi \colon \Gamma, \varphi \hookrightarrow \Gamma$ is the inclusion. So, such a partial element u is given by a family of elements $u\rho \in A\rho$ for each $\rho \in \Gamma(I)$ such that $\varphi\rho = 1$, satisfying $(u\rho)f = u(\rho f)$ whenever $f \colon J \to I$. Each $u \in \mathrm{Ter}(\Gamma; A)$ gives rise to the partial element $u\iota \in \mathrm{Ter}(\Gamma, \varphi; A\iota)$; a partial element is *extensible* if it is induced by such an element of $\mathrm{Ter}(\Gamma; A)$.

For the next definition note that if $A \in \mathrm{Ty}(\Gamma)$, then $\rho \in \Gamma(I)$ corresponds to $\rho \colon \mathbf{y}\, I \to \Gamma$ and thus $A\rho \in \mathrm{Ty}(\mathbf{y}\, I)$; also, any $\varphi \in \mathbb{F}(I)$ corresponds to $\varphi \in \mathrm{Ter}(\mathbf{y}\, I; \mathbb{F})$.

Definition 6.8.3. A *composition structure* for $A \in \mathrm{Ty}(\Gamma)$ is given by the following operations. For each I, $i \notin I$, $\rho \in \Gamma(I, i)$, $\varphi \in \mathbb{F}(I)$, u a partial element of $A\rho$ of extent φ, and $a_0 \in A\rho(i0)$ with $a_0 f = u_{(i0)f}$ for all $f \colon J \to I$ with

$\varphi f = 1_{\mathbb{F}}$ (i.e., $a_0\iota_\varphi = u(i0)$ if a_0 is considered as element of $\mathrm{Ter}(\mathbf{y}\, I; A\rho(i0))$), we require

$$\mathtt{comp}(I, i, \rho, \varphi, u, a_0) \in A\rho(i1)$$

such that for any $f\colon J \to I$ and $j \notin J$,

$$(\mathtt{comp}(I, i, \rho, \varphi, u, a_0))f = \mathtt{comp}(J, j, \rho(f, i = j), \varphi f, u(f, i = j), a_0 f),$$

and $\mathtt{comp}(I, i, \rho, 1_{\mathbb{F}}, u, a_0) = u_{(i1)}$.

A type $A \in \mathrm{Ty}(\Gamma)$ together with a composition structure $\mathtt{comp}$ on A is called a *fibrant type*, written $(A, \mathtt{comp}) \in \mathrm{FTy}(\Gamma)$. We will usually simply write $A \in \mathrm{FTy}(\Gamma)$ and $\mathtt{comp}_A$ for its composition structure. But observe that $A \in \mathrm{Ty}(\Gamma)$ can have different composition structures. Call a cubical set Γ *fibrant* if it is a fibrant type when Γ considered as type $\Gamma \in \mathrm{Ty}(\top)$ is fibrant where $\top$ is a terminal cubical set. A prime example of a fibrant cubical set is the singular cubical set of a topological space (see Appendix 6.C).

Theorem 6.8.4. *The CwF on cubical sets supporting dependent products, dependent sums, and natural numbers described above can be extended to fibrant types.*

Proof. For example, if $A \in \mathrm{FTy}(\Gamma)$ and $\sigma\colon \Delta \to \Gamma$, we set

$$\mathtt{comp}_{A\sigma}(I, i, \rho, \varphi, u, a_0) = \mathtt{comp}_A(I, i, \sigma\rho, \varphi, u, a_0)$$

as the composition structure on $A\sigma$ in $\mathrm{FTy}(\Delta)$. Type formers are treated analogously to their syntactic counterpart given in Section 6.4. Note that one also has to check that all equations between types are also preserved by their associated composition structures. □

Note that we can also, like in the syntax, define a composition structure on $\mathtt{Path}_A(u, v)$ given that A has one.

Semantic Glueing

Next we will give a semantic counterpart to the Glue construction. To define the semantic glueing as an element of $\mathrm{Ty}(\Gamma)$ it is not necessary that the given types have composition structures or that the functions are equivalences; this is only needed later to give the composition structure. Assume $\varphi \in \mathrm{Ter}(\Gamma; \mathbb{F})$, $T \in \mathrm{Ty}(\Gamma, \varphi)$, $A \in \mathrm{Ty}(\Gamma)$, and $w \in \mathrm{Ter}(\Gamma, \varphi; T \to A\iota)$ (where $A \to B$ is $\Pi(A, B\mathsf{p})$).

Definition 6.8.5. The *semantic glueing* $\mathtt{Glue}_\Gamma(\varphi, T, A, w) \in \mathrm{Ty}(\Gamma)$ is defined as follows. For $\rho \in \Gamma(I)$, we let $u \in \mathtt{Glue}(\varphi, T, A, w)\rho$ iff either

- $u \in T\rho$ and $\varphi\rho = 1_{\mathbb{F}}$; or
- $u = \mathtt{glue}(\varphi\rho, t, a)$ and $\varphi\rho \neq 1_{\mathbb{F}}$, where $t \in \mathrm{Ter}(\mathbf{y}\, I, \varphi\rho; T\rho)$ and $a \in \mathrm{Ter}(\mathbf{y}\, I; A\rho)$ such that $\mathtt{app}(w\rho, t) = a\iota \in \mathrm{Ter}(\mathbf{y}\, I, \varphi\rho; A\rho\iota)$.

For $f\colon J \to I$ we define the restriction uf of $u \in \mathtt{Glue}(\varphi, T, A, w)$ to be given by the restriction of $T\rho$ in the first case; in the second case, i.e., if $\varphi\rho \neq 1_{\mathbb{F}}$, we let $uf = \mathtt{glue}(\varphi\rho, t, a)f = t_f \in T\rho f$ in case $\varphi\rho f = 1_{\mathbb{F}}$, and otherwise $uf = \mathtt{glue}(\varphi\rho f, tf, af)$.

Here $\mathtt{glue}$ was defined as a constructor; we extend $\mathtt{glue}$ to any t in $\mathrm{Ter}(\mathbf{y}\, I; T\rho)$, a in $\mathrm{Ter}(\mathbf{y}\, I; A\rho)$ such that $\mathtt{app}(w\rho, t) = a$ (so if $\varphi\rho = 1_{\mathbb{F}}$) by $\mathtt{glue}(1_{\mathbb{F}}, t, a) = t_{\mathrm{id}_I}$. This way any element of $\mathtt{Glue}(\varphi, T, A, w)\rho$ is of the form $\mathtt{glue}(\varphi\rho, t, a)$ for suitable t and a, and restriction is given by

$$(\mathtt{glue}(\varphi\rho, t, a))f = \mathtt{glue}(\varphi\rho f, tf, af).$$

Note that we get

$$\begin{aligned} \mathtt{Glue}_\Gamma(1_{\mathbb{F}}, T, A, w) &= T \text{ and} \\ (\mathtt{Glue}_\Gamma(\varphi, T, A, w))\sigma &= \mathtt{Glue}_\Delta(\varphi\sigma, T\sigma, A\sigma, w\sigma) \end{aligned} \tag{6.3}$$

for $\sigma\colon \Delta \to \Gamma$. We define $\mathtt{unglue}(\varphi, w) \in \mathrm{Ter}(\Gamma.\, \mathtt{Glue}(\varphi, T, A, w); A\mathtt{p})$ by

$$\begin{aligned} \mathtt{unglue}(\varphi, w)(\rho, t) &= \mathtt{app}(w\rho, t)_{\mathrm{id}_I} \in A\rho && \text{whenever } \varphi\rho = 1_{\mathbb{F}}, \text{ and} \\ \mathtt{unglue}(\varphi, w)(\rho, \mathtt{glue}(\varphi, t, a)) &= a && \text{otherwise,} \end{aligned}$$

where $\rho \in \Gamma(I)$.

Definition 6.8.6. For $A, B \in \mathrm{Ty}(\Gamma)$ and $w \in \mathrm{Ter}(\Gamma; A \to B)$ an *equivalence structure* for w is given by the following operations such that for each

- $\rho \in \Gamma(I)$,
- $\varphi \in \mathbb{F}(I)$,
- $b \in B\rho$, and
- partial elements a of $A\rho$ and ω of $\mathtt{Path}_B(\mathtt{app}(w\rho, a), b\iota)\rho$ with extent φ,

we are given

- $\mathtt{e}_0(\rho, \varphi, b, a, \omega) \in A\rho$, and
- a path $\mathtt{e}_1(\rho, \varphi, b, a, \omega)$ between $\mathtt{app}(w\rho, \mathtt{e}_0(\rho, \varphi, b, a, \omega))$ and b

such that $\mathtt{e}_0(\rho, \varphi, b, a, \omega)\iota = a$, $\mathtt{e}_1(\rho, \varphi, b, a, \omega)\iota = \omega$ (where $\iota\colon \mathbf{y}\, I, \varphi \to \mathbf{y}\, I$) and for any $f\colon J \to I$ and $\nu = 0, 1$:

$$(\mathtt{e}_\nu(\rho, \varphi, b, a, \omega))f = \mathtt{e}_\nu(\rho f, \varphi f, bf, af, \omega f).$$

Following the argument in the syntax we can use the equivalence structure to explain a composition for $\mathtt{Glue}$.

Theorem 6.8.7. *If $A \in \mathrm{FTy}(\Gamma)$, $T \in \mathrm{FTy}(\Gamma, \varphi)$, and we have an equivalence structure for w, then we have a composition structure for $\mathtt{Glue}(\varphi, T, A, w)$ such that the equations* (6.3) *also hold for the respective composition structures.*

Semantic Universes

Assuming a Grothendieck universe of small sets in our ambient set theory, we can define $A \in \mathrm{Ty}_0(\Gamma)$ iff all $A\rho$ are small for $\rho \in \Gamma(I)$; and $A \in \mathrm{FTy}_0(\Gamma)$ iff $A \in \mathrm{Ty}_0(\Gamma)$ when forgetting the composition structure of A.

Definition 6.8.8. The semantic universe $\mathtt{U}$ is the cubical set defined by $\mathtt{U}(I) = \mathrm{FTy}_0(\mathbf{y}\, I)$; restriction along $f\colon J \to I$ is simply substitution along $\mathbf{y}\, f$.

We can consider $\mathtt{U}$ as an element of $\mathrm{Ty}(\Gamma)$. As such we can, as in the syntactic counterpart, define a composition structure on $\mathtt{U}$ using semantic glueing, so that $\mathtt{U} \in \mathrm{FTy}(\Gamma)$. Note that semantic glueing preserves smallness.

For $T \in \mathrm{Ter}(\Gamma; \mathtt{U})$ we can define decoding $\mathsf{El}\, T \in \mathrm{FTy}_0(\Gamma)$ by $(\mathsf{El}\, T)\rho = (T\rho)\,\mathrm{id}_I$ and likewise for the composition structure. For $A \in \mathrm{FTy}_0(\Gamma)$ we get its code $\ulcorner A \urcorner \in \mathrm{Ter}(\Gamma; \mathtt{U})$ by setting $\ulcorner A \urcorner \rho \in \mathrm{FTy}_0(\mathbf{y}\, I)$ to be given by the sets $(\ulcorner A \urcorner \rho) f = A(\rho f)$ and likewise for restrictions and composition structure. These operations satisfy $\mathsf{El} \ulcorner A \urcorner = A$ and $\ulcorner \mathsf{El}\, T \urcorner = T$.

6.8.3 Interpretation of the Syntax

Following [75] we define a partial interpretation function from raw syntax to the CwF with fibrant types given in the previous section.

To interpret the universe rules à la Russell we assume two Grothendieck universes in the underlying set theory, say *tiny* and *small* sets. So that any tiny set is small, and the set of tiny sets is small. For a cubical set X we define $\mathrm{FTy}_0(X)$ and $\mathrm{FTy}_1(X)$ as in the previous section, now referring to tiny and small sets, respectively. We get semantic universes $\mathtt{U}_i(I) = \mathrm{FTy}_i(\mathbf{y}\, I)$ for $i = 0, 1$; we identify those with their lifts to types. As noted above, these lifts carry a composition structure, and thus are fibrant. We also have $\mathtt{U}_0 \subseteq \mathtt{U}_1$ and thus $\mathrm{Ter}(X; \mathtt{U}_0) \subseteq \mathrm{Ter}(X; \mathtt{U}_1)$. Note that coding and decoding are, as set-theoretic operations, the same for both universes. We get that $\ulcorner \mathtt{U}_0 \urcorner \in \mathrm{Ter}(X; \mathtt{U}_1)$ which will serve as the interpretation of U.

In what follows, we define a partial interpretation function of raw syntax: $[\![\Gamma]\!]$, $[\![\Gamma; t]\!]$, and $[\![\Delta; \sigma]\!]$ by recursion on the raw syntax. Since we want to interpret a universe à la Russell we cannot assume terms and types to have different syntactic categories. The definition is given below and should be read such that the interpretation is defined whenever all interpretations on the right-hand sides are defined *and* make sense; so, e.g., for $[\![\Gamma]\!].\, \mathsf{El}\, [\![\Gamma; A]\!]$ below, we require that $[\![\Gamma]\!]$ is defined and a cubical set, $[\![\Gamma; A]\!]$ is defined, and $\mathsf{El}\, [\![\Gamma; A]\!] \in \mathrm{FTy}([\![\Gamma]\!])$. The interpretation for raw contexts is given by:

$$
\begin{array}{lll}
[\![()]\!] = \top & [\![\Gamma, x : A]\!] = [\![\Gamma]\!].\mathsf{El}\, [\![\Gamma; A]\!] & \text{if } x \notin \mathrm{dom}(\Gamma) \\
[\![\Gamma, \varphi]\!] = [\![\Gamma]\!], [\![\Gamma; \varphi]\!] & [\![\Gamma, i : \mathbb{I}]\!] = [\![\Gamma]\!].\mathbb{I} & \text{if } i \notin \mathrm{dom}(\Gamma)
\end{array}
$$

where $\top$ is a terminal cubical set and in the last equation $\mathbb{I}$ is considered as an element of $\mathrm{Ty}([\![\Gamma]\!])$. When defining $[\![\Gamma; t]\!]$ we require that $[\![\Gamma]\!]$ is defined and a cubical set; then $[\![\Gamma; t]\!]$ is a (partial) family of sets $[\![\Gamma; t]\!](I, \rho)$ for $I \in \mathcal{C}$ and

$\rho \in [\![\Gamma]\!](I)$ (leaving I implicit in the definition). We define:

$$\begin{aligned}
[\![\Gamma; \mathsf{U}]\!] &= \ulcorner \mathtt{U}_0 \urcorner \in \mathrm{Ter}([\![\Gamma]\!]; \mathtt{U}_1) \\
[\![\Gamma; \mathsf{N}]\!] &= \ulcorner \mathtt{N} \urcorner \in \mathrm{Ter}([\![\Gamma]\!]; \mathtt{U}_0) \\
[\![\Gamma; (x : A) \to B]\!] &= \ulcorner \Pi_{[\![\Gamma]\!]}(\mathsf{El}\,[\![\Gamma; A]\!], \mathsf{El}\,[\![\Gamma, x : A; B]\!]) \urcorner \\
[\![\Gamma; (x : A) \times B]\!] &= \ulcorner \Sigma_{[\![\Gamma]\!]}(\mathsf{El}\,[\![\Gamma; A]\!], \mathsf{El}\,[\![\Gamma, x : A; B]\!]) \urcorner \\
[\![\Gamma; \mathsf{Path}\ A\ a\ b]\!] &= \ulcorner \mathtt{Path}^{[\![\Gamma]\!]}_{\mathsf{El}\,[\![\Gamma; A]\!]}([\![\Gamma; a]\!], [\![\Gamma; b]\!]) \urcorner \\
[\![\Gamma; \mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A]\!] &= \ulcorner \mathtt{Glue}_{[\![\Gamma]\!]}([\![\Gamma; \varphi]\!], \mathsf{El}\,[\![\Gamma, \varphi; T]\!], \mathsf{El}\,[\![\Gamma; A]\!], [\![\Gamma, \varphi; f]\!]) \urcorner \\
[\![\Gamma; \lambda x : A.t]\!] &= \lambda_{[\![\Gamma]\!]; \mathsf{El}\,[\![\Gamma; A]\!]}([\![\Gamma, x : A; t]\!]) \\
[\![\Gamma; t\ u]\!] &= \mathtt{app}([\![\Gamma; t]\!], [\![\Gamma; u]\!]) \\
[\![\Gamma; \langle i \rangle\ t]\!] &= \langle\,\rangle_{[\![\Gamma]\!]}[\![\Gamma, i \colon \mathbb{I}; t]\!] \\
[\![\Gamma; t\ r]\!] &= [\![\Gamma; t]\!][\![\Gamma; r]\!]
\end{aligned}$$

where for path application, juxtaposition on the right-hand side is semantic path application. In the case of a bound variable, we assume that x (respectively i) is a *chosen* variable fresh for Γ; if this is not possible the expression is undefined. Moreover, all type formers should be read as those on fibrant types, i.e., also defining the composition structure. In the case of `Glue`, it is understood that the function part, i.e., the fourth argument of `Glue` in Definition 6.8.5 is $\mathsf{p} \circ [\![\Gamma, \varphi; f]\!]$ and the required (by Theorem 6.8.7) equivalence structure is to be extracted from $\mathsf{q} \circ [\![\Gamma, \varphi; f]\!]$ as in Section 6.5.3. In virtue of the remark in Section 6.8.2 we don't need type annotations to interpret applications. Note that coding and decoding tacitly refer to $[\![\Gamma]\!]$ as well. For the rest of the raw terms we also assume we are given $\rho \in [\![\Gamma]\!](I)$. Variables are interpreted by:

$$[\![\Gamma, x : A; x]\!]\rho = \mathsf{q}(\rho) \qquad [\![\Gamma, x : A; y]\!]\rho = [\![\Gamma; y]\!](\mathsf{p}(\rho)) \qquad [\![\Gamma, \varphi; y]\!]\rho = [\![\Gamma; y]\!]\rho$$

These should also be read to include the case when x or y are name variables; if x is a name variable, we require A to be $\mathbb{I}$. The interpretations of $[\![\Gamma; r]\!]\rho$ where r is not a name and $[\![\Gamma; \varphi]\!]\rho$ follow inductively as elements of $\mathbb{I}$ and $\mathbb{F}$, respectively.

Constants for dependent sums are interpreted by:

$$[\![\Gamma; (t, u)]\!]\rho = ([\![\Gamma; t]\!]\rho, [\![\Gamma; u]\!]\rho) \qquad [\![\Gamma; t.1]\!]\rho = \mathsf{p}([\![\Gamma; t]\!]\rho) \qquad [\![\Gamma; t.2]\!]\rho = \mathsf{q}([\![\Gamma; t]\!]\rho)$$

Likewise, constants for N will be interpreted by their semantic analogues (omitted). The interpretations for the constants related to glueing are

$$\begin{aligned}
[\![\Gamma; \mathsf{glue}\,[\varphi \mapsto t]\ u]\!]\rho &= \mathtt{glue}([\![\Gamma; \varphi]\!]\rho, [\![\Gamma, \varphi; t]\!]\hat{\rho}, [\![\Gamma; u]\!]\rho) \\
[\![\Gamma; \mathsf{unglue}\,[\varphi \mapsto f]\ u]\!]\rho &= \mathtt{unglue}([\![\Gamma; \varphi]\!], \mathsf{p} \circ [\![\Gamma; f]\!])(\rho, [\![\Gamma; u]\!]\rho)
\end{aligned}$$

where $[\![\Gamma, \varphi; t]\!]\hat{\rho}$ is the family assigning $[\![\Gamma, \varphi; t]\!](\rho f)$ to $J \in \mathcal{C}$ and $f \colon J \to I$ (and ρf refers to the restriction given by $[\![\Gamma]\!]$ which is assumed to be a cubical set). Partial elements are interpreted by

$$[\![\Gamma; [\ \varphi_1\ u_1, \ldots, \varphi_n\ u_n\ ]]\!]\rho = [\![\Gamma, \varphi_i; u_i]\!]\rho \qquad \text{if } [\![\Gamma; \varphi_i]\!]\rho = 1_{\mathbb{F}},$$

where for this to be defined we additionally assume that all $[\![\Gamma, \varphi_i; u_i]\!]$ are defined and $[\![\Gamma, \varphi_i; u_i]\!]\rho' = [\![\Gamma, \varphi_j; u_j]\!]\rho'$ for each $\rho' \in [\![\Gamma]\!](I)$ with $[\![\Gamma; \varphi_i \wedge \varphi_j]\!]\rho' = 1_{\mathbb{F}}$.

Finally, the interpretation of composition is given by

$$[\![\Gamma; \mathsf{comp}^i\ A\ [\varphi \mapsto u]\ a_0]\!]\rho = \mathtt{comp}_{\mathsf{El}\,[\![\Gamma, i:\mathbb{I}; A]\!]}(I, j, \rho', [\![\Gamma; \varphi]\!]\rho, [\![\Gamma, \varphi, i : \mathbb{I}; u]\!]\rho', [\![\Gamma; a_0]\!]\rho)$$

if $i \notin \mathrm{dom}(\Gamma)$, and where j is fresh and $\rho' = (\rho \mathrm{s}_j, i = j)$ with $\mathrm{s}_j : I, j \to I$ induced from the inclusion $I \subseteq I, j$.

The interpretation of raw substitutions $[\![\Delta; \sigma]\!]$ is a (partial) family of sets $[\![\Delta; \sigma]\!](I, \rho)$ for $I \in \mathcal{C}$ and $\rho \in [\![\Delta]\!](I)$. We set

$$[\![\Delta; ()]\!]\rho = *, \qquad [\![\Delta; (\sigma, x/t)]\!]\rho = ([\![\Delta; \sigma]\!]\rho, [\![\Delta; t]\!]\rho) \quad \text{if } x \notin \mathrm{dom}(\sigma),$$

where $*$ is the unique element of $\top(I)$. This concludes the definition of the interpretation of syntax.

In the following α stands for either a raw term or raw substitution. In the latter case, $\alpha\sigma$ denotes composition of substitutions.

Lemma 6.8.9. *Let Γ' be like Γ but with some φ's inserted, and assume both $[\![\Gamma]\!]$ and $[\![\Gamma']\!]$ are defined; then:*

1. *$[\![\Gamma']\!]$ is a sub-cubical set of $[\![\Gamma]\!]$;*
2. *if $[\![\Gamma; \alpha]\!]$ is defined, then so is $[\![\Gamma'; \alpha]\!]$ and they agree on $[\![\Gamma']\!]$.*

Lemma 6.8.10 (Weakening)**.** *Let $[\![\Gamma]\!]$ be defined.*

1. *If $[\![\Gamma, x : A, \Delta]\!]$ is defined, then so is $[\![\Gamma, x : A, \Delta; x]\!]$ which is moreover the projection to the x-part.*[5]
2. *If $[\![\Gamma, \Delta]\!]$ is defined, then so is $[\![\Gamma, \Delta; \mathrm{id}_\Gamma]\!]$ which is moreover the projection to the Γ-part.*
3. *If $[\![\Gamma, \Delta]\!]$, $[\![\Gamma; \alpha]\!]$ are defined and the variables in Δ are fresh for α, then $[\![\Gamma, \Delta; \alpha]\!]$ is defined and for $\rho \in [\![\Gamma, \Delta]\!](I)$:*

$$[\![\Gamma; \alpha]\!]([\![\Gamma, \Delta; \mathrm{id}_\Gamma]\!]\rho) = [\![\Gamma, \Delta; \alpha]\!]\rho$$

Lemma 6.8.11 (Substitution)**.** *For $[\![\Gamma]\!]$,$[\![\Delta]\!]$, $[\![\Delta; \sigma]\!]$, and $[\![\Gamma; \alpha]\!]$ defined with $\mathrm{dom}(\Gamma) = \mathrm{dom}(\sigma)$ (as lists), also $[\![\Delta; \alpha\sigma]\!]$ is defined and for $\rho \in [\![\Delta]\!](I)$:*

$$[\![\Gamma; \alpha]\!]([\![\Delta; \sigma]\!]\rho) = [\![\Delta; \alpha\sigma]\!]\rho$$

Lemma 6.8.12. *If $[\![\Gamma]\!]$ is defined and a cubical set, and $[\![\Gamma; \alpha]\!]$ is defined, then $([\![\Gamma; \alpha]\!]\rho)f = [\![\Gamma; \alpha]\!](\rho f)$.*

[5] E.g., if Γ is $y : B, z : C$, the projection to the x-part maps $(b, (c, (a, \delta)))$ to a, and the projection to the Γ-part maps $(b, (c, \delta))$ to (b, c).

To state the next theorem let us set $[\![\Gamma; \mathbb{I}]\!] = \ulcorner \mathbb{I} \urcorner$ and $[\![\Gamma; \mathbb{F}]\!] = \ulcorner \mathbb{F} \urcorner$ as elements of $\mathrm{Ty}_0([\![\Gamma]\!])$.

Theorem 6.8.13 (Soundness)**.** *We have the following implications, and all occurrences of* $[\![-]\!]$ *in the conclusions are defined. In* (3) *and* (5) *we allow* A *to be* $\mathbb{I}$ *or* $\mathbb{F}$*.*

1. *if* $\Gamma \vdash$ *, then* $[\![\Gamma]\!]$ *is a cubical set;*
2. *if* $\Gamma \vdash A$*, then* $[\![\Gamma; A]\!] \in \mathrm{Ter}([\![\Gamma]\!]; \mathtt{U}_1)$*;*
3. *if* $\Gamma \vdash t : A$*, then* $[\![\Gamma; t]\!] \in \mathrm{Ter}([\![\Gamma]\!]; \mathsf{El}\,[\![\Gamma; A]\!])$*;*
4. *if* $\Gamma \vdash A = B$*, then* $[\![\Gamma; A]\!] = [\![\Gamma; B]\!]$*;*
5. *if* $\Gamma \vdash a = b : A$*, then* $[\![\Gamma; a]\!] = [\![\Gamma; b]\!]$*;*
6. *if* $\Gamma \vdash \sigma : \Delta$*, then* $[\![\Gamma; \sigma]\!]$ *restricts to a natural transformation* $[\![\Gamma]\!] \to [\![\Delta]\!]$*.*

6.9 Extensions: Identity Types and Higher Inductive Types

In this section we consider possible extensions to cubical type theory. The first is an identity type defined using path types whose elimination principle holds as a judgmental equality. The second are two examples of higher inductive types.

6.9.1 Identity Types

We can use the path type to represent equalities. Using the composition operation, we can indeed build a substitution function $P(a) \to P(b)$ from any path between a and b. However, since we don't have in general the judgmental equality $\mathsf{transp}^i\ A\ a_0 = a_0$ if A is independent of i (which is an equality that we cannot expect geometrically in general, as shown in Appendix 6.C), this substitution function does not need to be the constant function when the path is constant. This means that, as in the previous model [15, 50], we don't get an interpretation of Martin-Löf identity type [59] with the standard judgmental equalities.

However, we can define another type which *does* give an interpretation of this identity type following an idea of Andrew Swan.

Identity Types

The basic idea of $\mathsf{Id}\ A\ a_0\ a_1$ is to define it in terms of $\mathsf{Path}\ A\ a_0\ a_1$ but also mark the paths where they are known to be constant. Formally, the formation and introduction rules are

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash a_0 : A \qquad \Gamma \vdash a_1 : A}{\Gamma \vdash \mathsf{Id}\ A\ a_0\ a_1} \qquad \frac{\Gamma \vdash \omega : \mathsf{Path}\ A\ a_0\ a_1[\varphi \mapsto \langle i \rangle\ a_0]}{\Gamma \vdash (\omega, \varphi) : \mathsf{Id}\ A\ a_0\ a_1}$$

and we can define $\mathsf{r}\,a = (1_a, 1_{\mathbb{F}}) : \mathsf{Id}\ A\ a\ a$ for $a : A$. The elimination rule, given $\Gamma \vdash a : A$, is

$$\frac{\begin{array}{c}\Gamma, x : A, \alpha : \mathsf{Id}\ A\ a\ x \vdash C \\ \Gamma \vdash d : C(x/a, \alpha/\mathsf{r}\,a) \qquad \Gamma \vdash b : A \qquad \Gamma \vdash \beta : \mathsf{Id}\ A\ a\ b\end{array}}{\Gamma \vdash \mathsf{J}_{x,\alpha.C}\ d\ b\ \beta : C(x/b, \alpha/\beta)}$$

together with the following judgmental equality in case β is of the form (ω, φ)

$$\mathsf{J}\ d\ b\ \beta = \mathsf{comp}^i\ C(x/\omega\ i, \alpha/\beta^*(i))\ [\varphi \mapsto d]\ d$$

where $\Gamma, i : \mathbb{I} \vdash \beta^*(i) : \mathsf{Id}\ A\ a\ (\omega\ i)$ is given by

$$\beta^*(i) = (\langle j \rangle\ \omega\ (i \wedge j), \varphi \vee (i = 0)).$$

Note that with this definition we get $\mathsf{J}\ d\ a\ (\mathsf{r}\,a) = d$ as desired.

The composition operation for Id is explained as follows. Given

$$\begin{array}{c}\Gamma, i : \mathbb{I} \vdash \mathsf{Id}\ A\ a_0\ a_1, \Gamma, \varphi, i : \mathbb{I} \vdash (\omega, \psi) : \mathsf{Id}\ A\ a_0\ a_1, \text{ and} \\ \Gamma \vdash (\omega_0, \psi_0) : (\mathsf{Id}\ A\ a_0\ a_1)(i0)[\varphi \mapsto (\omega(i0), \psi(i0))]\end{array}$$

we have the judgmental equality

$$\begin{array}{l}\mathsf{comp}^i\ (\mathsf{Id}\ A\ a_0\ a_1)\ [\varphi \mapsto (\omega, \psi)]\ (\omega_0, \psi_0) = \\ \qquad\qquad (\mathsf{comp}^i\ (\mathsf{Path}\ A\ a_0\ a_1)\ [\varphi \mapsto \omega]\ \omega_0, \varphi \wedge \psi(i1)).\end{array}$$

It can then be shown that the types $\mathsf{Id}\ A\ a\ b$ and $\mathsf{Path}\ A\ a\ b$ are (Path)-equivalent. In particular, a type is (Path)-contractible if, and only if, it is (Id)-contractible. The univalence axiom, proved in Section 6.7.2 for the Path-type, hence holds as well for the Id-type.[6]

Cofibration-Trivial Fibration Factorization

The same idea can be used to factorize an arbitrary map of (not necessary fibrant) cubical sets $f : A \to B$ into a cofibration followed by a trivial fibration. We define a "trivial fibration" to be a first projection from a total space of a contractible family of types and a "cofibration" to be a map that has the left lifting property against any trivial fibration. For this we define, for $b : B$, the type $T_f(b)$ to be the type of elements $[\varphi \mapsto a]$ with $\varphi \vdash a : A$ and $\varphi \vdash f\ a = b : B$.

Theorem 6.9.1. *The type $T_f(b)$ is contractible and the map*

$$A \to (b : B) \times T_f(b), \qquad a \longmapsto (f\ a, [1_{\mathbb{F}} \mapsto a])$$

is a cofibration.

The definition of the identity type can be seen as a special case of this, if we take the B the type of paths in A and for f the constant path function.

[6] This has been formally verified using the HASKELL implementation:
`https://github.com/mortberg/cubicaltt/blob/v1.0/examples/idtypes.ctt`

6.9.2 Higher Inductive Types

In this section we consider the extension of cubical type theory with two different higher inductive types: spheres and propositional truncation. The presentation in this section is syntactical, but it can be directly translated into semantic definitions.

Extension to Dependent Path Types

In order to formulate the elimination rules for higher inductive types, we need to extend the path type to *dependent path type*, which is described by the following rules. If $i : \mathbb{I} \vdash A$ and $\vdash a_0 : A(i0),\ a_1 : A(i1)$, then $\vdash \mathsf{Path}^i\ A\ a_0\ a_1$. The introduction rule is that $\vdash \langle i \rangle\ t : \mathsf{Path}^i\ A\ t(i0)\ t(i1)$ if $i : \mathbb{I} \vdash t : A$. The elimination rule is $\vdash p\ r : A(i/r)$ if $\vdash p : \mathsf{Path}^i\ A\ a_0\ a_1$ with equalities $p\ 0 = a_0 : A(i0)$ and $p\ 1 = a_1 : A(i1)$.

Spheres

We define the circle, S^1, by the rules:

$$\frac{\Gamma \vdash}{\Gamma \vdash \mathsf{S}^1} \qquad \frac{\Gamma \vdash}{\Gamma \vdash \mathsf{base} : \mathsf{S}^1} \qquad \frac{\Gamma \vdash r : \mathbb{I}}{\Gamma \vdash \mathsf{loop}(r) : \mathsf{S}^1}$$

with the equalities $\mathsf{loop}(0) = \mathsf{loop}(1) = \mathsf{base}$.

Since we want to represent the *free* type with one base point and a loop, we add composition as a *constructor* operation hcomp^i:

$$\frac{\Gamma, \varphi, i : \mathbb{I} \vdash u : \mathsf{S}^1 \qquad \Gamma \vdash u_0 : \mathsf{S}^1[\varphi \mapsto u(i0)]}{\Gamma \vdash \mathsf{hcomp}^i\ [\varphi \mapsto u]\ u_0 : \mathsf{S}^1}$$

with the equality $\mathsf{hcomp}^i\ [1_{\mathbb{F}} \mapsto u]\ u_0 = u(i1)$.

Given a dependent type $x : \mathsf{S}^1 \vdash A$ and $a : A(x/\mathsf{base})$ and a path $l : \mathsf{Path}^i\ A(x/\mathsf{loop}(i))\ a\ a$ we can define a function $g : (x : \mathsf{S}^1) \to A$ by the equations[7] $g\ \mathsf{base} = a$ and $g\ \mathsf{loop}(r) = l\ r$ and

$$g\ (\mathsf{hcomp}^i\ [\varphi \mapsto u]\ u_0) = \mathsf{comp}^i\ A(x/v)\ [\varphi \mapsto g\ u]\ (g\ u_0)$$

where $v = \mathsf{fill}^i\ \mathsf{S}^1\ [\varphi \mapsto u]\ u_0 = \mathsf{hcomp}^j\ [\varphi \mapsto u(i/i \wedge j), (i = 0) \mapsto u_0]\ u_0$.

This definition is non ambiguous since $l\ 0 = l\ 1 = a$.

We have a similar definition for the n-sphere, S^n, taking as constructors base and $\mathsf{loop}(r_1, \ldots, r_n)$.

[7] For the equation $g\ \mathsf{loop}(r) = l\ r$, it may be that l and r are dependent on the same name i, and we could not have followed this definition in the framework of [15].

Propositional Truncation

We define the propositional truncation, $\mathsf{inh}\ A$, of a type A by the rules:

$$\frac{\Gamma \vdash A}{\Gamma \vdash \mathsf{inh}\ A} \qquad \frac{\Gamma \vdash a : A}{\Gamma \vdash \mathsf{inc}\ a : \mathsf{inh}\ A}$$

$$\frac{\Gamma \vdash u_0 : \mathsf{inh}\ A \qquad \Gamma \vdash u_1 : \mathsf{inh}\ A \qquad \Gamma \vdash r : \mathbb{I}}{\Gamma \vdash \mathsf{squash}(u_0, u_1, r) : \mathsf{inh}\ A}$$

with the equalities $\mathsf{squash}(u_0, u_1, 0) = u_0$ and $\mathsf{squash}(u_0, u_1, 1) = u_1$.

As before, we add composition as a constructor, but only in the form[8]

$$\frac{\Gamma, \varphi, i : \mathbb{I} \vdash u : \mathsf{inh}\ A \qquad \Gamma \vdash u_0 : \mathsf{inh}\ A[\varphi \mapsto u(i0)]}{\Gamma \vdash \mathsf{hcomp}^i\ [\varphi \mapsto u]\ u_0 : \mathsf{inh}\ A}$$

with the equality $\mathsf{hcomp}^i\ [1_{\mathbb{F}} \mapsto u]\ u_0 = u(i1)$.

This provides only a definition of $\mathsf{comp}^i\ (\mathsf{inh}\ A)\ [\varphi \mapsto u]\ u_0$ in the case where A is independent of i, and we have to explain how to define the general case.

In order to do this, we define first two operations

$$\frac{\Gamma, i : \mathbb{I} \vdash A \qquad \Gamma \vdash u_0 : \mathsf{inh}\ A(i0)}{\Gamma \vdash \mathsf{transp}\ u_0 : \mathsf{inh}\ A(i1)}$$

$$\frac{\Gamma, i : \mathbb{I} \vdash A \qquad \Gamma, i : \mathbb{I} \vdash u : \mathsf{inh}\ A}{\Gamma \vdash \mathsf{squeeze}^i\ u : \mathsf{Path}\ (\mathsf{inh}\ A(i1))\ (\mathsf{transp}\ u(i0))\ u(i1)}$$

by the equations

$$\begin{array}{lcl}
\mathsf{transp}\ (\mathsf{inc}\ a) & = & \mathsf{inc}\ (\mathsf{comp}^i\ A\ []\ a) \\
\mathsf{transp}\ (\mathsf{squash}(u_0, u_1, r)) & = & \mathsf{squash}(\mathsf{transp}\ u_0, \mathsf{transp}\ u_1, r) \\
\mathsf{transp}\ (\mathsf{hcomp}^j\ [\varphi \mapsto u]\ u_0) & = & \mathsf{hcomp}^j\ [\varphi \mapsto \mathsf{transp}\ u]\ (\mathsf{transp}\ u_0) \\
 & & \\
\mathsf{squeeze}^i\ (\mathsf{inc}\ a) & = & \langle i \rangle\ \mathsf{inc}\ (\mathsf{comp}^j\ A(i \vee j) \\
 & & \qquad [(i = 1) \mapsto a(i1)]\ a) \\
\mathsf{squeeze}^i\ (\mathsf{squash}(u_0, u_1, r)) & = & \langle k \rangle\ \mathsf{squash}(\mathsf{squeeze}^i\ u_0\ k, \mathsf{squeeze}^i\ u_1\ k, \\
 & & \qquad r(i/k)) \\
\mathsf{squeeze}^i\ (\mathsf{hcomp}^j\ [\varphi \mapsto u]\ v) & = & \langle k \rangle\ \mathsf{hcomp}^j\ S\ (\mathsf{squeeze}^i\ v\ k)
\end{array}$$

where S is the system

$$[\delta \mapsto \mathsf{squeeze}^i\ u\ k,\ \varphi(i/k) \wedge (k = 0) \mapsto \mathsf{transp}\ u(i0),\ \varphi(i/k) \wedge (k = 1) \mapsto u(i1)]$$

and $\delta = \forall i.\varphi$, using Lemma 6.4.1.

[8] This restriction on the constructor is essential for the justification of the elimination rule below.

Using these operations, we can define the general composition

$$\frac{\Gamma, i : \mathbb{I} \vdash A \qquad \Gamma, \varphi, i : \mathbb{I} \vdash u : \mathsf{inh}\ A \qquad \Gamma \vdash u_0 : \mathsf{inh}\ A(i0)[\varphi \mapsto u(i0)]}{\Gamma \vdash \mathsf{comp}^i\ (\mathsf{inh}\ A)\ [\varphi \mapsto u]\ u_0 : \mathsf{inh}\ A(i1)[\varphi \mapsto u(i1)]}$$

by $\Gamma \vdash \mathsf{comp}^i\ (\mathsf{inh}\ A)\ [\varphi \mapsto u]\ u_0 = \mathsf{hcomp}^j\ [\varphi \mapsto \mathsf{squeeze}^i\ u\ j]\ (\mathsf{transp}\ u_0) : \mathsf{inh}\ A(i1)$.

Given $\Gamma \vdash B$ and $\Gamma \vdash q : (x\ y : B) \to \mathsf{Path}\ B\ x\ y$ and $f : A \to B$ we define $g : \mathsf{inh}\ A \to B$ by the equations:

$$\begin{array}{lcl} g\ (\mathsf{inc}\ a) & = & f\ a \\ g\ (\mathsf{squash}(u_0, u_1, r)) & = & q\ (g\ u_0)\ (g\ u_1)\ r \\ g\ (\mathsf{hcomp}^j\ [\varphi \mapsto u]\ u_0) & = & \mathsf{comp}^j\ B\ [\varphi \mapsto g\ u]\ (g\ u_0) \end{array}$$

6.10 Related and Future Work

Cubical ideas have proved useful to reason about equality in homotopy type theory [56]. In cubical type theory these techniques could be simplified as there are new judgmental equalities and better notations for manipulating higher dimensional cubes. Indeed some simple experiments using the Haskell implementation have shown that we can simplify some constructions in synthetic homotopy theory.[9]

Other approaches to extending intensional type theory with extensionality principles can be found in [4, 72]. These approaches have close connections to techniques for internalizing parametricity in type theory [13]. Further, nominal extensions to λ-calculus and semantical ideas related to the ones presented in this paper have recently also proved useful for justifying type theory with internalized parametricity [12].

The paper [37] provides a general framework for analyzing the uniformity condition, which applies to simplicial and cubical sets.

Large parts of the semantics presented in this paper have been formally verified in NuPrl by Mark Bickford[10], in particular, the definition of Kan filling in terms of composition as in Section 6.4.4 and composition for glueing as given in Section 6.6.2.

Following the usual reducibility method, we expect it to be possible to adapt our presheaf semantics to a proof of normalization and decidability of type checking. A first step in this direction is the proof of canonicity in [51]. We end the paper with a list of open problems and conjectures:

1. Extend the semantics of identity types to the semantics of inductive families.

2. Give a general syntax and semantics of higher inductive types.

[9] For details see: `https://github.com/mortberg/cubicaltt/tree/master/examples/`
[10] For details see: `http://www.nuprl.org/wip/Mathematics/cubical!type!theory/`

3. Extend the system with resizing rules and show normalization.

Acknowledgments. This work originates from discussions between the four authors around an implementation of a type system corresponding to the model described in [15]. This implementation indicated a problem with the representation of higher inductive types, e.g., the elimination rule for the circle, and suggested the need of extending this cubical model with a diagonal operation. The general framework (uniformity condition, connections, semantics of spheres and propositional truncation) is due to the second author. In particular, the glueing operation with its composition was introduced as a generalization of the operation described in [15] transforming an equivalence into a path, and with the condition $A = \mathsf{Glue}\ []\ A$. In a first attempt, we tried to force "regularity", i.e., the equation $\mathsf{transp}\ i\ A\ a_0 = a_0$ if A is independent of i (which seemed to be necessary in order to get filling from compositions, and which implies $\mathsf{Path} = \mathsf{Id}$). There was a problem however for getting regularity for the universe, that was discovered by Dan Licata (from discussions with Carlo Angiuli and Bob Harper). Thanks to this discovery, it was realized that regularity is actually not needed for the model to work. In particular, the third author noticed that we can remove the condition $A = \mathsf{Glue}\ []\ A$, and together with the last author, they derived the univalence axiom from the glueing operation as presented in the appendix. This was surprising since glueing was introduced a priori only as a way to transform equivalences into paths, but was later explained by a remark of Dan Licata (also presented in the appendix: we get univalence as soon as the transport map associated to this path is path equal to the given equivalence). The second author introduced then the restriction operation Γ, φ on contexts, which, as noticed by Christian Sattler, can be seen as an explicit syntax for the notion of cofibration, and designed the other proof of univalence in Section 6.7.2 from discussions between Nicola Gambino, Peter LeFanu Lumsdaine, and the third author. Not having regularity, the type of paths is not the same as the Id type but, as explained in Section 6.9.1, we can recover the usual identity type from the path type, following an idea of Andrew Swan.

The authors would like to thank the referees and Martín Escardó, Georges Gonthier, Dan Grayson, Peter Hancock, Dan Licata, Peter LeFanu Lumsdaine, Christian Sattler, Andrew Swan, Vladimir Voevodsky for many interesting discussions and remarks.

Appendix

6.A Details of Composition for Glueing

We build the element $\Gamma \vdash b_1 = \mathsf{comp}^i\ B\ [\psi \mapsto b]\ b_0 : (\mathsf{Glue}\ [\varphi \mapsto (T, f)]\ A)(i1)$ as the element $\mathsf{glue}\ [\varphi(i1) \mapsto t_1]\ a_1$ where:

$$\Gamma, \varphi(i1) \vdash t_1 : T(i1)[\psi \mapsto b(i1)]$$
$$\Gamma \vdash a_1 : A(i1)[\varphi(i1) \mapsto f(i1)\ t_1, \psi \mapsto (\mathsf{unglue}\ b)(i1)]$$

As intermediate steps, we gradually build elements that satisfy more and more of the equations that the final elements t_1 and a_1 should satisfy. The construction of these is given in five steps.

Before explaining how we can define them and why they are well defined, we illustrate the construction in Figure 6.6, with $\psi = (j = 1)$ and $\varphi = (i = 0) \vee (j = 1) \vee (i = 1)$.

We pose $\delta = \forall i.\varphi$ (cf. Section 6.3), so that we have that δ is independent from i, and in our example $\delta = (j = 1)$ and it represents the right-hand side of the picture.

1. The element $a'_1 : A(i1)$ is a first approximation of a_1, but a'_1 is not necessarily in the image of $f(i1)$ in $\Gamma, \varphi(i1)$;

2. the partial element $\delta \vdash t'_1 : T(i1)$, which is a partial final result for $\varphi(i1) \vdash t_1$;

3. the partial path $\delta \vdash \omega$, between a'_1 and the image of t'_1;

4. both the final element $\varphi(i1) \vdash t_1$ and a path $\varphi(i1) \vdash \alpha$ between a'_1 and $f(i1)\ t_1$;

5. finally, we build a_1 from a'_1 and α.

We define:

$$\Gamma, \psi, i : \mathbb{I} \vdash a = \mathsf{unglue}\ b : A[\varphi \mapsto f\ b]$$
$$\Gamma \vdash a_0 = \mathsf{unglue}\ b_0 : A(i0)[\varphi(i0) \mapsto f(i0)\ b_0, \psi \mapsto a(i0)]$$

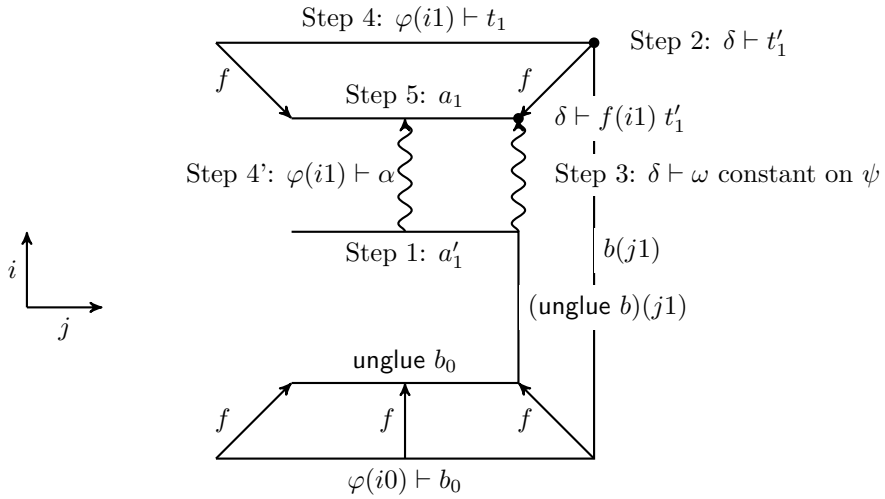


Figure 6.6: Composition for glueing

Step 1 We define a'_1 as the composition of a and $\mathsf{unglue}\ b_0$, in the direction i,

$$\Gamma \vdash a'_1 = \mathsf{comp}^i\ A\ [\psi \mapsto\ a]\ a_0 : A(i1)[\psi \mapsto\ a(i1)] \tag{6.4}$$

which is well defined since $\mathsf{unglue}\ b_0 = (\mathsf{unglue}\ b)(i0)$ over the extent ψ.

Step 2 We also define t'_1 as the composition of b and b_0, in the direction i:

$$\Gamma, \delta \vdash t'_1 = \mathsf{comp}^i\ T\ [\psi \mapsto\ b]\ b_0 : T(i1)[\psi \mapsto\ b(i1)] \tag{6.5}$$

which is well defined because

$$\begin{cases} \Gamma, \delta, i : \mathbb{I}, \psi \vdash b : T & \text{by Lemma 6.4.2,} \\ \Gamma, \delta \vdash b_0 : T(i0)[\psi \mapsto b(i0)] & \text{as } \delta \leqslant \varphi(i0). \end{cases}$$

Moreover, since

$$\begin{cases} \Gamma, \delta, \psi, i : \mathbb{I} \vdash a = f\ b & \text{by } \delta \leqslant \varphi, \\ \Gamma, \delta \vdash\ a_0 = f(i0)\ b_0 & \text{by } \delta \leqslant \varphi(i0), \end{cases}$$

we can re-express a'_1 on the extent δ:

$$\Gamma, \delta \vdash a'_1 = \mathsf{comp}^i\ A\ [\psi \mapsto f\ b]\ (f(i0)\ b_0)$$

Step 3 We can hence find a path ω connecting a'_1 and $f(i1)\ t'_1$ in Γ, δ using Lemma 6.5.2:

$$\Gamma, \delta \vdash \omega = \mathsf{pres}^i\ f\ [\psi \mapsto b]\ b_0 : (\mathsf{Path}\ A(i1)\ a'_1\ \ (f(i1)\ t'_1))\ [\psi \mapsto \langle j \rangle\ a(i1)]$$

Picking a fresh name j, we have

$$\Gamma, \delta, j : \mathbb{I} \vdash \omega\ j : A(i1)[(j0) \mapsto a'_1, (j1) \mapsto f(i1)\ t'_1, \psi \mapsto a(i1)]. \tag{6.6}$$

Step 4 Now we define the final element t_1 as the inverse image of a'_1 by $f(i1)$, together with the path α between a'_1 and $f(i1)\ t_1$, in $\Gamma, \varphi(i1) \vdash$, using Lemma 6.5.3:

$$\Gamma, \varphi(i1) \vdash (t_1, \alpha) = \mathsf{equiv}\ f(i1)\ [\delta \mapsto\ (t'_1, \omega), \psi \mapsto\ \ (b(i1), \langle j \rangle\ a'_1)]\ a'_1$$

with

$$\begin{cases} \Gamma, \varphi(i1) \vdash t_1 : T(i1)[\delta \mapsto\ t'_1, \psi \mapsto\ b(i1)], \\ \Gamma, \varphi(i1) \vdash \alpha : (\mathsf{Path}\ A(i1)\ a'_1\ \ (f(i1)\ t_1))\ [\delta \mapsto \omega, \psi \mapsto\ \langle j \rangle\ a'_1)]. \end{cases}$$

These are well defined because the two systems in δ and ψ are compatible:

$$\begin{cases} \Gamma, \delta, \psi \vdash t'_1 = b(i1) & \text{by (6.5)}, \\ \Gamma, \delta, \psi \vdash \omega = \langle j \rangle\ a'_1 & \text{by (6.6) and (6.4)}. \end{cases}$$

Picking a fresh name j, we have:

$$\begin{aligned} \Gamma, \varphi(i1), j : \mathbb{I} \vdash \alpha\ j : A(i1)[(j = 0) \mapsto a'_1,\ (j = 1) \mapsto f(i1)\ t_1, \\ \delta \mapsto\ a'_1,\ \psi \mapsto a(i1)] \end{aligned} \tag{6.7}$$

Step 5 Finally, we define a_1 by composition of α and a'_1:

$$\begin{aligned} \Gamma \vdash\ a_1 := \mathsf{comp}^j\ A(i1)\ [\varphi(i1) \mapsto \alpha\ j, \psi \mapsto\ a(i1)]\ a'_1 : \\ A(i1)[\varphi(i1) \mapsto \alpha\ 1, \psi \mapsto\ a(i1)] \end{aligned}$$

which is well defined because

$$\begin{cases} \Gamma, j : \mathbb{I}, \varphi(i1), \psi \vdash \alpha\ j = a(i1) & \text{by (6.7)}, \\ \Gamma, \varphi(i1) \vdash \alpha\ 0 = a'_1 & \text{by (6.7)}, \\ \Gamma, \psi \vdash a(i1) = a'_1 & \text{by (6.4)}, \end{cases}$$

and since $\Gamma, \varphi(i1) \vdash \alpha\ 1 = f(i1)\ t_1$, we can re-express the type of a_1 in the following way:

$$\Gamma \vdash a_1 : A(i1)[\varphi(i1) \mapsto f(i1)\ t_1,\ \psi \mapsto a(i1)]$$

Which is exactly what we needed to build $\Gamma \vdash b_1 := \mathsf{glue}\ [\varphi(i1) \mapsto t_1]\ a_1 : B(i1)[\psi \mapsto b(i1)]$.

Finally we check that $b_1 = \mathsf{comp}^i\ T\ [\psi \mapsto\ b]\ b_0$ on δ:

$$\begin{aligned} b_1 &= \mathsf{glue}\ [\varphi(i1) \mapsto t_1]\ a_1 && \text{by def.} \\ &= t_1 : T(i1)[\delta \mapsto\ t'_1, \psi \mapsto\ b(i1)] && \text{as } \varphi(i1) = 1_{\mathbb{F}} \\ &= t'_1 && \text{as } \delta = 1_{\mathbb{F}} \\ &= \mathsf{comp}^i\ T\ [\psi \mapsto\ b]\ b_0 && \text{by def.} \end{aligned}$$

6.B Univalence from Glueing

We also give two alternative proofs of the univalence axiom for Path only involving the glue construction.[11] The first is a direct proof of the standard formulation of the univalence axiom while the second goes through an alternative formulation as in Corollary 6.7.3.[12]

Lemma 6.B.1. *For $\Gamma \vdash A : \mathsf{U}$, $\Gamma \vdash B : \mathsf{U}$, and an equivalence $\Gamma \vdash f :$ Equiv $A\ B$ we have the following constructions:*

1. $\Gamma \vdash \mathsf{eqToPath}\ f : \mathsf{Path}\ \mathsf{U}\ A\ B$;
2. $\Gamma \vdash \mathsf{Path}\ (A \to B)\ (\mathsf{transp}^i(\mathsf{eqToPath}\ f\ i))\ f.1$ *is inhabited; and*
3. *if $f = \mathsf{equiv}^i(P\ i)$ for $\Gamma \vdash P : \mathsf{Path}\ \mathsf{U}\ A\ B$, then the following type is inhabited:*

$$\Gamma \vdash \mathsf{Path}\ (\mathsf{Path}\ \mathsf{U}\ A\ B)\ (\mathsf{eqToPath}\ (\mathsf{equiv}^i(P\ i)))\ P$$

Proof. For (1) we define

$$\mathsf{eqToPath}\ f = \langle i \rangle\ \mathsf{Glue}\ [(i = 0) \mapsto (A, f), (i = 1) \mapsto (B, \mathsf{equiv}^k B)]\ B. \tag{6.8}$$

Note that here $\mathsf{equiv}^k B$ is an equivalence between B and B (see Section 6.7.1). For (2) we have to closely look at how the composition was defined for Glue. By unfolding the definition, we see that the left-hand side of the equality is equal $f.1$ composed with multiple transports in a constant type; using filling and functional extensionality, these transports can be shown to be equal to the identity; for details see the formal proof.

The term for (3) is given by:

$$\begin{aligned} \langle j \rangle\ \langle i \rangle\ \mathsf{Glue}\ [&(i = 0) \mapsto (A, \mathsf{equiv}^k(P\ k)), \\ &(i = 1) \mapsto (B, \mathsf{equiv}^k B), \\ &(j = 1) \mapsto (P\ i, \mathsf{equiv}^k(P(i \vee k)))] \\ &B \end{aligned}$$

□

[11] The proofs of the univalence axiom have all been formally verified inside the system using the Haskell implementation. We note that the proof of Theorem 6.7.2 can be given such that it extends $f.2$ and hence in Corollary 6.7.3 we don't need the fact that isEquiv $X\ A\ f.1$ is a proposition. For details see: `https://github.com/mortberg/cubicaltt/blob/v1.0/examples/univalence.ctt`

[12] The second of these proofs is inspired by a proof by Dan Licata from: `https://groups.google.com/d/msg/homotopytypetheory/j2KBIvDw53s/YTDK4D0NFQAJ`

Corollary 6.B.2 (Univalence axiom)**.** *For the canonical map*

$$\mathsf{pathToEq} : (A\,B : \mathsf{U}) \to \mathsf{Path}\,\mathsf{U}\,A\,B \to \mathsf{Equiv}\;A\;B$$

we have that $\mathsf{pathToEq}\,A\,B$ *is an equivalence for all* $A : \mathsf{U}$ *and* $B : \mathsf{U}$*.*

Proof 1. Let us first show that the canonical map $\mathsf{pathToEq}$ is path equal to:

$$\mathsf{equiv} = \lambda A\,B : \mathsf{U}.\,\lambda P : \mathsf{Path}\,\mathsf{U}\,A\,B.\,\mathsf{equiv}^i(P\,i)$$

By function extensionality, it suffices to check this pointwise. Using path-induction, we may assume that P is reflexivity. In this case $\mathsf{pathToEq}\,A\,A\,1_A$ is the identity equivalence by definition. Because being an equivalence is a proposition, it thus suffices that the first component of $\mathsf{equiv}^i A$ is propositionally equal to the identity. By definition, this first component is given by transport (now in the constant type A) which is easily seen to be the identity using filling (see Section 6.4.4).

Thus it suffices to prove that $\mathsf{equiv}\,A\,B$ is an equivalence. To do so it is enough to give an inverse (see Theorems 4.2.3 and 4.2.6 of [79]). But $\mathsf{eqToPath}$ is a left inverse by Lemma 6.B.1 (3), and a right inverse by Lemma 6.B.1 (2) using that being an equivalence is a proposition. □

Proof 2. Points (1) and (2) of Lemma 6.B.1 imply that $\mathsf{Equiv}\;A\;B$ is a retract of $\mathsf{Path}\,\mathsf{U}\,A\,B$. Hence $(X : \mathsf{U}) \times \mathsf{Equiv}\;A\;X$ is a retract of $(X : \mathsf{U}) \times \mathsf{Path}\,\mathsf{U}\,A\,X$. But $(X : \mathsf{U}) \times \mathsf{Path}\,\mathsf{U}\,A\,X$ is contractible, so $(X : \mathsf{U}) \times \mathsf{Equiv}\;A\;X$ is also contractible as a retract of a contractible type. As discussed in Section 6.7.2 this is an alternative formulation of the univalence axiom and the rest of this proof follows as there. □

Note that the first proof uses all three of the points of Lemma 6.B.1 while the second proof only uses the first two. As the second proof only uses the first two points it is possible to prove it if point (1) is defined as in Example 6.6.2 leading to a slightly simpler proof of point (2).

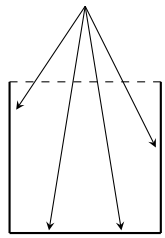
6.C Singular Cubical Sets

Recall the functor $\mathcal{C} \to \mathbf{Top}, I \mapsto [0,1]^I$ given at (6.1) in Section 6.8.1. In particular, the face maps $(ib) \colon I - i \to I$ (for $b = 0_{\mathbb{I}}$ or $1_{\mathbb{I}}$) induce the maps $(ib) \colon [0,1]^{I-i} \to [0,1]^I$ by $i(ib)u = b$ and $j(ib)u = ju$ if $j \neq i$ is in I. If ψ is in $\mathbb{F}(I)$ and u in $[0,1]^I$, then ψu is a truth value.

We assume given a family of idempotent functions $r_I : [0,1]^I \times [0,1] \to [0,1]^I \times [0,1]$ such that

1. $r_I(u,z) = (u,z)$ iff $\partial_I u = 1$ or $z = 0$, and
2. for any *strict* f in $\mathsf{Hom}(I,J)$ we have $r_J(f \times \mathrm{id})r_I = r_J(f \times \mathrm{id})$.

Such a family can for instance be defined as in the following picture ("retraction from above center"). If the center has coordinate $(1/2, 2)$, then $r_I(u,z) = r_I(u',z')$ is equivalent to $(2-z')(-1+2u) = (2-z)(-1+2u')$.



Property (1) holds for the retraction defined by this picture. The property (2) can be reformulated as $r_I(u,z) = r_I(u',z') \to r_J(fu,z) = r_J(fu',z')$. It also holds in this case, since $r_I(u,z) = r_I(u',z')$ is then equivalent to $(2-z')(-1+2u) = (2-z)(-1+2u')$, which implies $(2-z')(-1+2fu) = (2-z)(-1+2fu')$ if f is strict.

Using this family, we can define for each ψ in $\mathbb{F}(I)$ an idempotent function

$$r_\psi : [0,1]^I \times [0,1] \to [0,1]^I \times [0,1]$$

having for fixed-points the element (u,z) such that $\psi u = 1$ or $z = 0$. This function r_ψ is completely characterized by the following properties:

1. $r_\psi = \mathrm{id}$ if $\psi = 1$
2. $r_\psi = r_\psi r_I$ if $\psi \neq 1$
3. $r_\psi(u,z) = (u,z)$ if $z = 0$
4. $r_\psi((ib) \times \mathrm{id}) = ((ib) \times \mathrm{id}) r_{\psi(ib)}$

These properties imply for instance $r_{\partial_I}(u,z) = (u,z)$ if $\partial_I u = 1$ or $z = 0$ and so they imply $r_{\partial_I} = r_I$. They also imply that $r_\psi(u,z) = (u,z)$ if $\psi u = 1$.

From these properties, we can prove the uniformity of the family of functions r_ψ.

Theorem 6.C.1. *If f is in $\mathsf{Hom}(I,J)$ and ψ is in $\mathbb{F}(J)$, then $r_\psi(f \times \mathrm{id}) = (f \times \mathrm{id}) r_{\psi f}$.*

This is proved by induction on the number of element of I (the result being clear if I is empty).

A particular case is $r_J(f \times \mathrm{id}) = (f \times \mathrm{id}) r_{\partial_J f}$. Note that, in general, $\partial_J f$ is not ∂_I.

A direct consequence of the previous theorem is the following.

Corollary 6.C.2. *The singular cubical set associated to a topological space has a composition structure.*

Chapter 7

Canonicity for Cubical Type Theory

7.1 Introduction

Cubical type theory as presented in [26] is a dependent type theory which allows one to directly argue about n-dimensional cubes, and in which function extensionality and Voevodsky's Univalence Axiom [87] are provable. Cubical type theory is inspired by a *constructive* model of dependent type theory in cubical sets [26] and a previous variation thereof [15, 50]. One of its important ingredients is that expressions can depend on *names* to be thought of as ranging over a formal unit interval $\mathbb{I}$.

Even though the consistency of the calculus already follows from its model in cubical sets, desired—and expected—properties like normalization and decidability of type checking are not yet established. This note presents a first step in this direction by proving *canonicity* for natural numbers in the following form: given a context I of the form $i_1 : \mathbb{I}, \ldots, i_k : \mathbb{I}$, $k \geq 0$, and a derivation of $I \vdash u : \mathsf{N}$, there is a unique $n \in \mathbb{N}$ with $I \vdash u = \mathsf{S}^n\, 0 : \mathsf{N}$. This n can moreover be effectively calculated. Canonicity in this form also gives an alternative proof of the consistency of cubical type theory (see Corollary 7.4.22).

The main idea to prove canonicity is as follows. First, we devise an operational semantics given by a typed and deterministic weak-head reduction extending the judgmental equality of cubical type theory. This is given for general contexts although we later on will only use it on terms whose only free variables are name variables, i.e., variables of type $\mathbb{I}$. One result we obtain is that our reduction relation is "complete" in the sense that any term in a name context whose type is the natural numbers can be reduced to one in weak-head normal form (so to zero or a successor). Second, we will follow Tait's computability method [78, 62] and devise computability predicates on *typed* expressions in name contexts and corresponding computability relations (to interpret judgmental equality). These computability predicates are indexed

by the list of free name variables of the involved expressions and should be such that substitution induces a cubical set structure on them. This poses a major difficulty given that the reduction relation is in general not closed under name substitutions. A solution is to require for computability that reduction should behave "coherently" with substitution: simplified, reducing an expression and then substituting should be related, by the computability relation, to first substituting and then reducing. A similar condition appeared independently in the Computational Higher Type Theory of Angiuli, Harper, and Wilson [8] and Angiuli and Harper [7] who work in an untyped setting; they achieve similar results but for a theory not encompassing the Univalence Axiom.

In a way, our technique can be considered as a presheaf extension of the computability argument given in [2, 1]; the latter being an adaption of the former using a typed reduction relation instead. A similar extension of this technique has been used to show the independence of Markov's principle in type theory [31].

The rest of the paper is organized as follows. In Section 7.2 we introduce the typed reduction relation. Section 7.3 defines the computability predicates and relations and shows their important properties. In Section 7.4 we show that cubical type theory is sound w.r.t. the computability predicates; this entails canonicity. Section 7.5 sketches how to adapt the computability argument for the system extended with the circle. We conclude by summarizing and listing further work in the last section. We assume that the reader is familiar with cubical type theory as given in [26].

7.2 Reduction

In this section we give an operational semantics for cubical type theory in the form of a typed and deterministic weak-head reduction. Below we will introduce the relations $\Gamma \vdash A \succ B$ and $\Gamma \vdash u \succ v : A$. These relations are deterministic in the following sense: if $\Gamma \vdash A \succ B$ and $\Gamma \vdash A \succ C$, then B and C are equal as expressions (i.e., up to α-equivalence); and, if $\Gamma \vdash u \succ v : A$ and $\Gamma \vdash u \succ w : B$, then v and w are equal as expressions. Moreover, these relations extend judgmental equality, i.e., if $\Gamma \vdash A \succ B$, then $\Gamma \vdash A = B$, and if $\Gamma \vdash u \succ v : A$, then $\Gamma \vdash u = v : A$.

For a context $\Gamma \vdash$, a Γ-*introduced* expression is an expression whose outer form is an introduction, so one of the form

$$0, \mathsf{S}\, u, \mathsf{N}, \lambda x : A.u, (x : A) \to B, (u, v), (x : A) \times B, \mathsf{U}, \langle i \rangle u, \mathsf{Path}\, A\, u\, v,$$
$$[\varphi_1\ t_1, \ldots, \varphi_n\ t_n], \mathsf{glue}\, [\varphi \mapsto t]\, a, \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A,$$

where we require $\varphi \neq 1 \mod \Gamma$ (which we from now on write as $\Gamma \vdash \varphi \neq 1 : \mathbb{F}$) for the latter two cases, and in the case of a system (third to last) we require $\Gamma \vdash \varphi_1 \vee \cdots \vee \varphi_n = 1 : \mathbb{F}$ but $\Gamma \vdash \varphi_k \neq 1 : \mathbb{F}$ for each k. In case Γ only contains object and interval variable declarations (and *no* restrictions Δ, ψ) we simply

refer to Γ-introduced as *introduced*. In such a context, $\Gamma \vdash \varphi = \psi : \mathbb{F}$ iff $\varphi = \psi$ as elements of the face lattice $\mathbb{F}$; since $\mathbb{F}$ satisfies the disjunction property, i.e.,

$$\varphi \vee \psi = 1 \Rightarrow \varphi = 1 \text{ or } \psi = 1,$$

a system as above will never be introduced in such a context without restrictions. We call an expression *non-introduced* if it is not introduced and abbreviate this as "n.i." (often this is referred to as neutral or non-canonical). A Γ-introduced expression is normal w.r.t. $\Gamma \vdash \cdot \succ \cdot$ and $\Gamma \vdash \cdot \succ \cdot : A$.

We will now give the definition of the reduction relation starting with the rules concerning basic type theory.

$$\frac{\Gamma \vdash u \succ v : A \qquad \Gamma \vdash A = B}{\Gamma \vdash u \succ v : B}$$

$$\frac{\Gamma, x : \mathsf{N} \vdash C \qquad \Gamma \vdash z : C(x/0) \qquad \Gamma \vdash s : (x : \mathsf{N}) \to C \to C(x/\mathsf{S}\,x)}{\Gamma \vdash \mathsf{natrec}\,0\,z\,s \succ z : C(x/0)}$$

$$\frac{\Gamma \vdash t : \mathsf{N} \qquad \Gamma, x : \mathsf{N} \vdash C \qquad \Gamma \vdash z : C(x/0) \qquad \Gamma \vdash s : (x : \mathsf{N}) \to C \to C(x/\mathsf{S}\,x)}{\Gamma \vdash \mathsf{natrec}\,(\mathsf{S}\,t)\,z\,s \succ s\,t\,(\mathsf{natrec}\,t\,z\,s) : C(x/\mathsf{S}\,t)}$$

$$\frac{\Gamma \vdash t \succ t' : \mathsf{N} \qquad \Gamma, x : \mathsf{N} \vdash C \qquad \Gamma \vdash z : C(x/0) \qquad \Gamma \vdash s : (x : \mathsf{N}) \to C \to C(x/\mathsf{S}\,x)}{\Gamma \vdash \mathsf{natrec}\,t\,z\,s \succ \mathsf{natrec}\,t'\,z\,s : C(x/t')}$$

$$\frac{\Gamma, x : A \vdash t : B \qquad \Gamma \vdash u : A}{\Gamma \vdash (\lambda x : A.t)\,u \succ t(x/u) : B(x/u)} \qquad \frac{\Gamma \vdash t \succ t' : (x : A) \to B \qquad \Gamma \vdash u : A}{\Gamma \vdash t\,u \succ t'u : B(x/u)}$$

$$\frac{\Gamma, x : A \vdash B \qquad \Gamma \vdash u : A \qquad \Gamma \vdash v : B(x/u)}{\Gamma \vdash (u, v).1 \succ u : A} \qquad \frac{\Gamma \vdash t \succ t' : (x : A) \times B}{\Gamma \vdash t.1 \succ t'.1 : A}$$

$$\frac{\Gamma, x : A \vdash B \qquad \Gamma \vdash u : A \qquad \Gamma \vdash v : B(x/u)}{\Gamma \vdash (u, v).2 \succ v : B(x/u)} \qquad \frac{\Gamma \vdash t \succ t' : (x : A) \times B}{\Gamma \vdash t.2 \succ t'.2 : B(x/t'.1)}$$

Note, $\mathsf{natrec}\,t\,z\,s$ is not considered as an application (opposed to the presentation in [26]); also the order of the arguments is different to have the main premise as first argument.

Next, we give the reduction rules for Path-types. Note, that like for Π-types, there is no η-reduction or expansion, and also there is no reduction for the end-points of a path.

$$\frac{\Gamma \vdash A \qquad \Gamma, i : \mathbb{I} \vdash t : A \qquad \Gamma \vdash r : \mathbb{I}}{\Gamma \vdash (\langle i \rangle t)\,r \succ t(i/r) : A} \qquad \frac{\Gamma \vdash t \succ t' : \mathsf{Path}\,A\,u\,v \qquad \Gamma \vdash r : \mathbb{I}}{\Gamma \vdash t\,r \succ t'r : A}$$

The next rules concern reductions for Glue.

$$\frac{\Gamma \vdash A \qquad \Gamma, \varphi \vdash T \qquad \Gamma, \varphi \vdash w : \mathsf{Equiv}\, T\, A \qquad \Gamma \vdash \varphi = 1 : \mathbb{F}}{\Gamma \vdash \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A \succ T}$$

$$\frac{\begin{array}{c}\Gamma, \varphi \vdash w : \mathsf{Equiv}\, T\, A \\ \Gamma, \varphi \vdash t : T \qquad \Gamma \vdash a : A[\varphi \mapsto w.1\, t] \qquad \Gamma \vdash \varphi = 1 : \mathbb{F}\end{array}}{\Gamma \vdash \mathsf{glue}\, [\varphi \mapsto t]\, a \succ t : T}$$

$$\frac{\begin{array}{c}\Gamma, \varphi \vdash w : \mathsf{Equiv}\, T\, A \\ \Gamma, \varphi \vdash t : T \qquad \Gamma \vdash a : A[\varphi \mapsto w.1\, t] \qquad \Gamma \vdash \varphi \neq 1 : \mathbb{F}\end{array}}{\Gamma \vdash \mathsf{unglue}\, [\varphi \mapsto w]\, (\mathsf{glue}\, [\varphi \mapsto t]\, a) \succ a : A}$$

$$\frac{\Gamma, \varphi \vdash w : \mathsf{Equiv}\, T\, A \qquad \Gamma \vdash u : \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A \qquad \Gamma \vdash \varphi = 1 : \mathbb{F}}{\Gamma \vdash \mathsf{unglue}\, [\varphi \mapsto w]\, u \succ w.1\, u : A}$$

$$\frac{\Gamma \vdash u \succ u' : \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A \qquad \Gamma \vdash \varphi \neq 1 : \mathbb{F}}{\Gamma \vdash \mathsf{unglue}\, [\varphi \mapsto w]\, u \succ \mathsf{unglue}\, [\varphi \mapsto w]\, u' : A}$$

Note that in [26] the annotation $[\varphi \mapsto w]$ of unglue was left implicit. The rules for systems are given by:

$$\frac{\begin{array}{c}\Gamma \vdash \varphi_1 \vee \cdots \vee \varphi_n = 1 : \mathbb{F} \qquad \Gamma, \varphi_i \vdash A_i \ (1 \leq i \leq n) \\ \Gamma, \varphi_i \wedge \varphi_j \vdash A_i = A_j \ (1 \leq i, j \leq n) \qquad k \text{ minimal with } \Gamma \vdash \varphi_k = 1 : \mathbb{F}\end{array}}{\Gamma \vdash [\varphi_1\ A_1, \ldots, \varphi_n\ A_n] \succ A_k}$$

$$\frac{\begin{array}{c}\Gamma \vdash \varphi_1 \vee \cdots \vee \varphi_n = 1 : \mathbb{F} \qquad \Gamma \vdash A \qquad \Gamma, \varphi_i \vdash t_i : A \ (1 \leq i \leq n) \\ \Gamma, \varphi_i \wedge \varphi_j \vdash t_i = t_j : A \ (1 \leq i, j \leq n) \qquad k \text{ minimal with } \Gamma \vdash \varphi_k = 1 : \mathbb{F}\end{array}}{\Gamma \vdash [\varphi_1\ t_1, \ldots, \varphi_n\ t_n] \succ t_k : A}$$

The reduction rules for the universe are:

$$\frac{\Gamma \vdash A \succ B : \mathsf{U}}{\Gamma \vdash A \succ B}$$

$$\frac{\Gamma \vdash A : \mathsf{U} \qquad \Gamma, \varphi \vdash T : \mathsf{U} \qquad \Gamma, \varphi \vdash w : \mathsf{Equiv}\, T\, A \qquad \Gamma \vdash \varphi = 1 : \mathbb{F}}{\Gamma \vdash \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A \succ T : \mathsf{U}}$$

Finally, the reduction rules for compositions are given as follows.

$$\frac{\begin{array}{c}\Gamma, i : \mathbb{I} \vdash A \succ B \\ \Gamma \vdash \varphi : \mathbb{F} \qquad \Gamma, \varphi, i : \mathbb{I} \vdash u : A \qquad \Gamma \vdash u_0 : A(i0)[\varphi \mapsto u(i0)]\end{array}}{\Gamma \vdash \mathsf{comp}^i\, A\, [\varphi \mapsto u]\, u_0 \succ \mathsf{comp}^i\, B\, [\varphi \mapsto u]\, u_0 : B(i1)}$$

$$\frac{\Gamma \vdash \varphi : \mathbb{F} \qquad \Gamma, \varphi, i : \mathbb{I} \vdash u : \mathsf{N} \qquad \Gamma, \varphi, i : \mathbb{I} \vdash u = 0 : \mathsf{N}}{\Gamma \vdash \mathsf{comp}^i\, \mathsf{N}\, [\varphi \mapsto u]\, 0 \succ 0 : \mathsf{N}}$$

$$\frac{\Gamma \vdash \varphi : \mathbb{F} \qquad \Gamma, \varphi, i : \mathbb{I} \vdash u : \mathsf{N} \qquad \Gamma, \varphi, i : \mathbb{I} \vdash w : \mathsf{N} \quad \Gamma, \varphi, i : \mathbb{I} \vdash u = \mathsf{S}\, w : \mathsf{N} \qquad \Gamma \vdash u_0 : \mathsf{N} \qquad \Gamma, \varphi \vdash u(i0) = \mathsf{S}\, u_0 : \mathsf{N}}{\Gamma \vdash \mathsf{comp}^i\, \mathsf{N}\, [\varphi \mapsto u]\, (\mathsf{S}\, u_0) \succ \mathsf{S}(\mathsf{comp}^i\, \mathsf{N}\, [\varphi \mapsto \mathsf{pred}\, u]\, u_0) : \mathsf{N}}$$

Here pred is the usual predecessor function defined using natrec.[1]

$$\frac{\Gamma \vdash \varphi : \mathbb{F} \qquad \Gamma, \varphi, i : \mathbb{I} \vdash u : \mathsf{N} \qquad \Gamma \vdash u_0 : \mathsf{N}[\varphi \mapsto u(i0)] \qquad \Gamma \vdash u_0 \succ v_0 : \mathsf{N}}{\Gamma \vdash \mathsf{comp}^i\, \mathsf{N}\, [\varphi \mapsto u]\, u_0 \succ \mathsf{comp}^i\, \mathsf{N}\, [\varphi \mapsto u]\, v_0 : \mathsf{N}}$$

$$\frac{\Gamma \vdash \varphi : \mathbb{F} \qquad \Gamma, i : \mathbb{I} \vdash A \qquad \Gamma, i : \mathbb{I}, x : A \vdash B \quad \Gamma, \varphi, i : \mathbb{I} \vdash u : (x : A) \to B \qquad \Gamma \vdash u_0 : ((x : A) \to B)(i0)[\varphi \mapsto u(i0)]}{\begin{array}{c}\Gamma \vdash \mathsf{comp}^i\, ((x : A) \to B)\, [\varphi \mapsto u]\, u_0 \succ \\ \lambda y : A(i1).\mathsf{comp}^i\, B(x/\bar{y})\, [\varphi \mapsto u\, \bar{y}]\, (u_0\, \bar{y}(i0)) : (x : A(i1)) \to B(i1) \\ \text{where } y' = \mathsf{fill}^i\, A(i/1 - i)\, []\, y \text{ and } \bar{y} = y'(i/1 - i)\end{array}}$$

$$\frac{\Gamma \vdash \varphi : \mathbb{F} \qquad \Gamma, i : \mathbb{I} \vdash A \qquad \Gamma, i : \mathbb{I}, x : A \vdash B \quad \Gamma, \varphi, i : \mathbb{I} \vdash u : (x : A) \times B \qquad \Gamma \vdash u_0 : ((x : A) \times B)(i0)[\varphi \mapsto u(i0)]}{\begin{array}{c}\Gamma \vdash \mathsf{comp}^i\, ((x : A) \times B)\, [\varphi \mapsto u]\, u_0 \succ \\ (v(i1), \mathsf{comp}^i\, B(x/v)\, [\varphi \mapsto u.2]\, (u_0.2)) : (x : A(i1)) \times B(i1) \\ \text{where } v = \mathsf{fill}^i\, A\, [\varphi \mapsto u.1]\, (u_0.1)\end{array}}$$

$$\frac{\Gamma \vdash \varphi : \mathbb{F} \qquad \Gamma, i : \mathbb{I} \vdash A \qquad \Gamma, i : \mathbb{I} \vdash v : A \qquad \Gamma, i : \mathbb{I} \vdash w : A \quad \Gamma, \varphi, i : \mathbb{I} \vdash u : \mathsf{Path}\, A\, v\, w \qquad \Gamma \vdash u_0 : \mathsf{Path}\, A(i0)\, v(i0)\, w(i0)[\varphi \mapsto u(i0)]}{\begin{array}{c}\Gamma \vdash \mathsf{comp}^i\, (\mathsf{Path}\, A\, v\, w)\, [\varphi \mapsto u]\, u_0 \succ \\ \langle j \rangle\, \mathsf{comp}^i\, A\, [(j = 0) \mapsto v, (j = 1) \mapsto w, \varphi \mapsto u\, j]\, (u_0\, j) : \mathsf{Path}\, A(i1)\, v(i1)\, w(i1)\end{array}}$$

$$\frac{\Gamma, i : \mathbb{I} \vdash A \qquad \Gamma, i : \mathbb{I} \vdash \varphi : \mathbb{F} \qquad \Gamma, i : \mathbb{I}, \varphi \vdash T \quad \Gamma, i : \mathbb{I}, \varphi \vdash w : \mathsf{Equiv}\, T\, A \qquad \Gamma \vdash \psi : \mathbb{F} \qquad \Gamma, \psi, i : \mathbb{I} \vdash u : \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A \quad \Gamma \vdash u_0 : (\mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A)(i0)[\psi \mapsto u(i0)]}{\begin{array}{c}\Gamma \vdash \mathsf{comp}^i\, (\mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A)\, [\psi \mapsto u]\, u_0 \succ \\ \mathsf{glue}\, [\varphi(i1) \mapsto t_1]\, a_1 : (\mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A)(i1)\end{array}}$$

[1] We never have to reduce in the system of a composition when defining composition for natural numbers in this way, which also gives that reduction over Γ never refers to reduction in a restricted context Γ, φ (given that Γ is not restricted).

Here a_1 and t_1 are defined like in [26], i.e., given by

$$\begin{aligned}
a &= \mathsf{unglue}\,[\varphi \mapsto w]\,u && \Gamma, i : \mathbb{I}, \psi \\
a_0 &= \mathsf{unglue}\,[\varphi(i0) \mapsto w(i0)]\,u_0 && \Gamma \\
\delta &= \forall i.\varphi && \Gamma \\
a'_1 &= \mathsf{comp}^i\,A\,[\psi \mapsto a]\,a_0 && \Gamma \\
t'_1 &= \mathsf{comp}^i\,T\,[\psi \mapsto u]\,u_0 && \Gamma, \delta \\
\omega &= \mathsf{pres}^i\,w\,[\psi \mapsto u]\,u_0 && \Gamma, \delta \\
(t_1, \alpha) &= \mathsf{equiv}\,w(i1)\,[\delta \mapsto (t'_1, \omega), \psi \mapsto (u(i1), \langle j \rangle a'_1)]\,a'_1 && \Gamma, \varphi(i1) \\
a_1 &= \mathsf{comp}^j\,A(i1)\,[\varphi(i1) \mapsto \alpha\,j, \psi \mapsto a(i1)]\,a'_1 && \Gamma
\end{aligned}$$

where we indicated the intended context on the right.

$$\frac{\Gamma \vdash \varphi : \mathbb{F} \qquad \Gamma, \varphi, i : \mathbb{I} \vdash u : \mathsf{U} \qquad \Gamma \vdash u_0 : \mathsf{U}[\varphi \mapsto u(i0)]}{\Gamma \vdash \mathsf{comp}^i\,\mathsf{U}\,[\varphi \mapsto u]\,u_0 \succ \mathsf{Glue}\,[\varphi \mapsto (u(i1), \mathsf{equiv}^i\,u(i/1-i))]\,u_0 : \mathsf{U}}$$

Here equiv^i is defined as in [26]. This concludes the definition of the reduction relation.

For $\Gamma \vdash A$ we write $A!_\Gamma$ if there is B such that $\Gamma \vdash A \succ B$; in this case B is uniquely determined by A and we denote B by $A{\downarrow_\Gamma}$; if A is normal we set $A{\downarrow_\Gamma}$ to be A. Similarly for $\Gamma \vdash u : A$, $u!_\Gamma^A$ and $u{\downarrow_\Gamma^A}$. Note that if a term or type has a reduct it is non-introduced. We usually drop the subscripts and sometimes also superscripts since they can be inferred.

From now on we will mainly consider contexts $I, J, K, \ldots$ only built from dimension name declarations; so such a context is of the form $i_1 : \mathbb{I}, \ldots, i_n : \mathbb{I}$ for $n \geq 0$. We sometimes write I, i for $I, i : \mathbb{I}$. Substitutions between such contexts will be denoted by $f, g, h, \ldots$. The resulting category with such name contexts I as objects and substitutions $f \colon J \to I$ is reminiscent of the category of cubes as defined in [26, Section 8.1] with the difference that the names in a contexts I are ordered and not sets. This difference is not crucial for the definition of computability predicates in the next section but it simplifies notations. (Note that if I' is a permutation of I, then the substitution assigning to each name in I itself is an isomorphism $I' \to I$.) We write $r \in \mathbb{I}(I)$ if $I \vdash r : \mathbb{I}$, and $\varphi \in \mathbb{F}(I)$ if $I \vdash \varphi : \mathbb{F}$.

Note that in general reductions $I \vdash A \succ B$ or $I \vdash u \succ v : A$ are not closed under substitutions $f \colon J \to I$. For example, if u is a system $[(i = 0)\ u_1, 1\ u_2]$, then $i \vdash u \succ u_2 : A$ (assuming everything is well typed), but $\vdash u(i0) \succ u_1(i0) : A(i0)$ and u_1, u_2 might be chosen that $u_1(i0)$ and $u_2(i0)$ are judgmentally equal but not syntactically (and even normal by considering two λ-abstractions where the body is not syntactically but judgmentally equal). Another example is when u is $\mathsf{unglue}\,[\varphi \mapsto w]\,(\mathsf{glue}\,[\varphi \mapsto t]\,a)$ with $\varphi \neq 1$ and with $f \colon J \to I$ such that $\varphi f = 1$; then u reduces to a, but uf reduces to $wf.1\,(\mathsf{glue}\,[\varphi f \mapsto tf]\,af)$ which is in general not syntactically equal to af.

We write $I \vdash A \succ_{\mathsf{s}} B$ and $I \vdash u \succ_{\mathsf{s}} v : A$ if the respective reduction is closed under name substitutions. That is, $I \vdash A \succ_{\mathsf{s}} B$ whenever $J \vdash Af \succ Bf$ for all $f \colon J \to I$. Note that in the above definition, all the rules which do not have a premise with a *negated* equation in $\mathbb{F}$ and which do not have a premise referring to another reduction are closed under substitution.

7.3 Computability Predicates

In this section we define computability predicates and establish the properties we need for the proof of Soundness in the next section. We will define when a type is *computable* or *forced*, written $I \Vdash_\ell A$, when two types are forced equal, $I \Vdash_\ell A = B$, when an element is computable or forced, $I \Vdash_\ell u : A$, and when two elements are forced equal, $I \Vdash_\ell u = v : A$. Here ℓ is the *level* which is either 0 or 1, the former indicating smallness.

The definition is given as follows: by main recursion on ℓ (that is, we define "$\Vdash_0$" before "$\Vdash_1$") we define by induction-recursion [35]

$$\begin{aligned} &I \Vdash_\ell A \\ &I \Vdash_\ell A = B \\ &I \Vdash_\ell u : A && \text{by recursion on } I \Vdash_\ell A \\ &I \Vdash_\ell u = v : A && \text{by recursion on } I \Vdash_\ell A \end{aligned}$$

where the former two are mutually defined by induction, and the latter two mutually by recursion on the derivation of $I \Vdash_\ell A$. Formally, $I \Vdash_\ell A$ and $I \Vdash_\ell A = B$ are witnessed by derivations for which we don't introduce notations since the definitions of $I \Vdash_\ell u : A$ and $I \Vdash_\ell u = v : A$ don't depend on the derivation of $I \Vdash_\ell A$. Each such derivation has a height as an ordinal, and often we will employ induction not only on the structure of such a derivation but on its height.

Note that the arguments and definitions can be adapted to a hierarchy of universes by allowing ℓ to range over a (strict) well-founded poset.

We write $I \Vdash_\ell A \doteq B$ for the conjunction of $I \Vdash_\ell A$, $I \Vdash_\ell B$, and $I \Vdash_\ell A = B$. For $\varphi \in \mathbb{F}(I)$ we write $f \colon J \to I, \varphi$ for $f \colon J \to I$ with $\varphi f = 1$; furthermore we write

$$\begin{aligned} &I, \varphi \Vdash_\ell A && \text{for} && \forall f \colon J \to I, \varphi\, (J \Vdash_\ell Af) \;\&\; I, \varphi \vdash A, \\ &I, \varphi \Vdash_\ell A = B && \text{for} && \forall f \colon J \to I, \varphi\, (J \Vdash_\ell Af = Bf) \;\&\; I, \varphi \vdash A = B, \\ &I, \varphi \Vdash_\ell u : A && \text{for} && \forall f \colon J \to I, \varphi\, (J \Vdash_\ell uf : Af) \;\&\; I, \varphi \vdash u : A, \\ &I, \varphi \Vdash_\ell u = v : A && \text{for} && \forall f \colon J \to I, \varphi\, (J \Vdash_\ell uf = vf : Af) \;\& \\ & && && \qquad I, \varphi \vdash u = v : A. \end{aligned}$$

where the last two abbreviations need suitable premises to make sense. Note that $I, 1 \Vdash_\ell A$ is a priori stronger than $I \Vdash_\ell A$; that these notions are equivalent follows from the Monotonicity Lemma below. Moreover, the definition is such that $I \vdash \mathcal{J}$ whenever $I \Vdash_\ell \mathcal{J}$ (where $\mathcal{J}$ is any judgment form); it is shown

in Remark 7.4.18 that the condition $I, \varphi \vdash \mathcal{J}$ in the definition of $I, \varphi \Vdash_\ell \mathcal{J}$ above is actually not needed and follows from the other.

$\boxed{I \Vdash_\ell A}$ assuming $I \vdash A$ (i.e., the rules below all have a suppressed premise $I \vdash A$).

$$\frac{}{I \Vdash_\ell \mathsf{N}}\ \text{N-C}$$

$$\frac{I, 1 \Vdash_\ell A \qquad \forall f\colon J \to I \forall u (J \Vdash_\ell u : Af \Rightarrow J \Vdash_\ell B(f, x/u)) \\ \forall f\colon J \to I \forall u, v (J \Vdash_\ell u = v : Af \Rightarrow J \Vdash_\ell B(f, x/u) \doteqdot B(f, x/v))}{I \Vdash_\ell (x : A) \to B}\ \text{Pi-C}$$

$$\frac{I, 1 \Vdash_\ell A \qquad \forall f\colon J \to I \forall u (J \Vdash_\ell u : Af \Rightarrow J \Vdash_\ell B(f, x/u)) \\ \forall f\colon J \to I \forall u, v (J \Vdash_\ell u = v : Af \Rightarrow J \Vdash_\ell B(f, x/u) \doteqdot B(f, x/v))}{I \Vdash_\ell (x : A) \times B}\ \text{Si-C}$$

$$\frac{I, 1 \Vdash_\ell A \qquad I \Vdash_\ell a_0 : A \qquad I \Vdash_\ell a_1 : A}{I \Vdash_\ell \mathsf{Path}\, A\, a_0\, a_1}\ \text{Pa-C}$$

$$\frac{1 \neq \varphi \in \mathbb{F}(I) \qquad I, 1 \Vdash_\ell A \qquad I, \varphi \Vdash_\ell \mathsf{Equiv}\, T\, A \\ I, \varphi \Vdash_\ell w : \mathsf{Equiv}\, T\, A \qquad I, \varphi \Vdash_\ell \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A}{I \Vdash_\ell \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A}\ \text{Gl-C} \qquad \frac{}{I \Vdash_1 \mathsf{U}}\ \text{U-C}$$

$$\frac{A \text{ n.i.} \qquad \forall f\colon J \to I (Af! \ \&\ J \Vdash_\ell Af\downarrow) \\ \forall f\colon J \to I \forall g\colon K \to J (K \Vdash_\ell Af\downarrow g = Afg\downarrow)}{I \Vdash_\ell A}\ \text{Ni-C}$$

Note, that the rule Gl-C above is not circular, as for any $f\colon J \to I, \varphi$ we have $\varphi f = 1$ and so $(\mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A) f$ is non-introduced.

$\boxed{I \Vdash_\ell A = B}$ assuming $I \Vdash_\ell A$, $I \Vdash_\ell B$, and $I \vdash A = B$ (i.e., each rule below has the suppressed premises $I \Vdash_\ell A$, $I \Vdash_\ell B$, and $I \vdash A = B$).

$$\frac{}{I \Vdash_\ell \mathsf{N} = \mathsf{N}}\ \text{N-E}$$

$$\frac{I, 1 \Vdash_\ell A = A' \\ \forall f\colon J \to I \forall u (J \Vdash_\ell u : Af \Rightarrow J \Vdash_\ell B(f, x/u) = B'(f, x/u))}{I \Vdash_\ell (x : A) \to B = (x : A') \to B'}\ \text{Pi-E}$$

$$\frac{I, 1 \Vdash_\ell A = A' \\ \forall f\colon J \to I \forall u (J \Vdash_\ell u : Af \Rightarrow J \Vdash_\ell B(f, x/u) = B'(f, x/u))}{I \Vdash_\ell (x : A) \times B = (x : A') \times B'}\ \text{Si-E}$$

$$\frac{I, 1 \Vdash_\ell A = B \qquad I \Vdash_\ell a_0 = b_0 : A \qquad I \Vdash_\ell a_1 = b_1 : A}{I \Vdash_\ell \mathsf{Path}\, A\, a_0\, a_1 = \mathsf{Path}\, B\, b_0\, b_1}\ \textsc{Pa-E}$$

$$\frac{\begin{array}{c} 1 \neq \varphi \in \mathbb{F}(I) \qquad I, 1 \Vdash_\ell A = A' \\ I, \varphi \Vdash_\ell \mathsf{Equiv}\, T\, A = \mathsf{Equiv}\, T'\, A' \qquad I, \varphi \Vdash_\ell w = w' : \mathsf{Equiv}\, T\, A \\ I, \varphi \Vdash_\ell \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A = \mathsf{Glue}\, [\varphi \mapsto (T', w')]\, A' \end{array}}{I \Vdash_\ell \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A = \mathsf{Glue}\, [\varphi \mapsto (T', w')]\, A'}\ \textsc{Gl-E}$$

$$\frac{}{I \Vdash_1 \mathsf{U} = \mathsf{U}}\ \textsc{U-E} \qquad \frac{A \text{ or } B \text{ n.i.} \qquad \forall f \colon J \to I (J \Vdash_\ell Af{\downarrow} = Bf{\downarrow})}{I \Vdash_\ell A = B}\ \textsc{Ni-E}$$

$\boxed{I \Vdash_\ell u : A \text{ by induction on } I \Vdash_\ell A}$ assuming $I \vdash u : A$.

Case N-C.

$$\frac{}{I \Vdash_\ell 0 : \mathsf{N}} \qquad \frac{I \Vdash_\ell u : \mathsf{N}}{I \Vdash_\ell \mathsf{S}\, u : \mathsf{N}}$$

$$\frac{\begin{array}{c} u \text{ n.i.} \qquad \forall f \colon J \to I (uf!^{\mathsf{N}} \ \&\ J \Vdash_\ell uf{\downarrow}^{\mathsf{N}} : \mathsf{N}) \\ \forall f \colon J \to I \forall g \colon K \to J (K \Vdash_\ell uf{\downarrow}^{\mathsf{N}} g = ufg{\downarrow}^{\mathsf{N}} : \mathsf{N}) \end{array}}{I \Vdash_\ell u : \mathsf{N}}$$

Case Pi-C.

$$\frac{\begin{array}{c} \forall f \colon J \to I \forall u (J \Vdash_\ell u : Af \Rightarrow J \Vdash_\ell wf\, u : B(f, x/u)) \\ \forall f \colon J \to I \forall u, v (J \Vdash_\ell u = v : Af \Rightarrow J \Vdash_\ell wf\, u = wf\, v : B(f, x/u)) \end{array}}{I \Vdash_\ell w : (x : A) \to B}$$

Case Si-C.

$$\frac{I \Vdash_\ell u.1 : A \qquad I \Vdash_\ell u.2 : B(x/u.1)}{I \Vdash_\ell u : (x : A) \times B}$$

Case Pa-C.

$$\frac{\begin{array}{c} \forall f \colon J \to I \forall r \in \mathbb{I}(J) (J \Vdash_\ell uf\, r : Af) \\ I \Vdash_\ell u\, 0 = a_0 : A \qquad I \Vdash_\ell u\, 1 = a_1 : A \end{array}}{I \Vdash_\ell u : \mathsf{Path}\, A\, a_0\, a_1}$$

Case Gl-C.

$$\frac{\begin{array}{c} I, \varphi \Vdash_\ell u : \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A \\ \forall f \colon J \to I \forall w' (J, \varphi f \Vdash w' = wf : \mathsf{Equiv}\, Tf\, Af \Rightarrow \\ \qquad J \Vdash \mathsf{unglue}\, [\varphi f \mapsto w']\, uf = \mathsf{unglue}\, [\varphi f \mapsto wf]\, uf : Af) \end{array}}{I \Vdash_\ell u : \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A}$$

Later we will see that from the premises of Gl-C we get $I \Vdash w = w : \mathsf{Equiv}\, T\, A$, and the second premise above implies in particular $I \Vdash \mathsf{unglue}\, [\varphi \mapsto w]\, u : A$;

the quantification over other possible equivalences is there to ensure invariance for the annotation.

Case U-C.

$$\frac{I \Vdash_0 A}{I \Vdash_1 A : \mathsf{U}}$$

Case Ni-C.

$$\frac{\forall f\colon J \to I (J \Vdash_\ell u f : A f\downarrow)}{I \Vdash_\ell u : A}$$

$\boxed{I \Vdash_\ell u = v : A \text{ by induction on } I \Vdash_\ell A}$ assuming $I \Vdash_\ell u : A$, $I \Vdash_\ell v : A$, and $I \vdash u = v : A$. (I.e., each of the rules below has the suppressed premises $I \Vdash_\ell u : A$, $I \Vdash_\ell v : A$, and $I \vdash u = v : A$, but they are not arguments to the definition of the predicate. This is subtle since in, e.g., the rule for pairs we only know $I \Vdash_\ell v.2 : B(x/v.1)$ not $I \Vdash_\ell v.2 : B(x/u.1)$.)

Case N-C.

$$\frac{}{I \Vdash_\ell 0 = 0 : \mathsf{N}} \qquad \frac{I \Vdash_\ell u = v : \mathsf{N}}{I \Vdash_\ell \mathsf{S}\, u = \mathsf{S}\, v : \mathsf{N}}$$

$$\frac{u \text{ or } v \text{ n.i.} \qquad \forall f (J \Vdash_\ell u f\downarrow^{\mathsf{N}} = v f\downarrow^{\mathsf{N}} : \mathsf{N})}{I \Vdash_\ell u = v : \mathsf{N}}$$

Case Pi-C.

$$\frac{\forall f\colon J \to I \forall u (J \Vdash_\ell u : A f \Rightarrow J \Vdash_\ell w f\, u = w' f\, u : B(f, x/u))}{I \Vdash_\ell w = w' : (x : A) \to B}$$

Case Si-C.

$$\frac{I \Vdash_\ell u.1 = v.1 : A \qquad I \Vdash_\ell u.2 = v.2 : B(x/u.1)}{I \Vdash_\ell u = v : (x : A) \times B}$$

Case Pa-C.

$$\frac{\forall f\colon J \to I \forall r \in \mathbb{I}(J) (J \Vdash_\ell u f\, r = v f\, r : A f)}{I \Vdash_\ell u = v : \mathsf{Path}\, A\, a_0\, a_1}$$

Case Gl-C.

$$\frac{\begin{array}{c} I, \varphi \Vdash_\ell u = v : \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A \\ I, 1 \Vdash_\ell \mathsf{unglue}\, [\varphi \mapsto w]\, u = \mathsf{unglue}\, [\varphi \mapsto w]\, v : A \end{array}}{I \Vdash_\ell u = v : \mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A}$$

Case U-C.

$$\frac{I \Vdash_0 A = B}{I \Vdash_1 A = B : \mathsf{U}}$$

Case NI-C.

$$\frac{\forall f\colon J \to I(J \Vdash_\ell uf = vf : Af{\downarrow})}{I \Vdash_\ell u = v : A}$$

Note that the definition is such that $I \Vdash_\ell A = B$ implies $I \Vdash_\ell A$ and $I \Vdash_\ell B$; and, likewise, $I \Vdash_\ell u = v : A$ gives $I \Vdash_\ell u : A$ and $I \Vdash_\ell v : A$.

Remark 7.3.1.

1. In the rule NI-E and the rule for $I \Vdash_\ell u = v : \mathsf{N}$ in case u or v are non-introduced we suppressed the premise that the reference to "$\downarrow$" is actually well defined; it is easily seen that if $I \Vdash_\ell A$, then $A{\downarrow}$ is well defined, and similarly for $I \Vdash_\ell u : \mathsf{N}$, $u{\downarrow}^{\mathsf{N}}$ is well defined.

2. It follows from the substitution lemma below that $I \Vdash_\ell A$ whenever A is non-introduced and $I \vdash A \succ_{\mathsf{s}} B$ with $I \Vdash_\ell B$. (Cf. also the Expansion Lemma below.)

3. Note that once we also have proven transitivity, symmetry, and monotonicity, the last premise of NI-C (and similarly in the rule for non-introduced naturals) can be restated as $J \Vdash_\ell Af{\downarrow} = A{\downarrow}f$ for all $f\colon J \to I$.

Lemma 7.3.2. *The computability predicates are independent of the derivation, i.e., if we have two derivations trees d_1 and d_2 of $I \Vdash_\ell A$, then*

$$I \Vdash_\ell^{d_1} u : A \Leftrightarrow I \Vdash_\ell^{d_2} u : A, \text{ and}$$
$$I \Vdash_\ell^{d_1} u = v : A \Leftrightarrow I \Vdash_\ell^{d_2} u = v : A$$

where $\Vdash_\ell^{d_i}$ refers to the predicate induced by d_i.

Proof. By main induction on ℓ and a side induction on the derivations d_1 and d_2. Since the definition of $I \Vdash_\ell A$ is syntax directed both d_1 and d_2 are derived by the same rule. The claim thus follows from the IH. □

Lemma 7.3.3.

1. *If $I \Vdash_\ell A$, then $I \vdash A$ and:*
   (a) $I \Vdash_\ell u : A \Rightarrow I \vdash u : A$,
   (b) $I \Vdash_\ell u = v : A \Rightarrow I \vdash u = v : A$.
2. *If $I \Vdash_\ell A = B$, then $I \vdash A = B$.*

Lemma 7.3.4.

1. *If $I \Vdash_0 A$, then:*
   (a) $I \Vdash_1 A$
   (b) $I \Vdash_0 u : A \Leftrightarrow I \Vdash_1 u : A$
   (c) $I \Vdash_0 u = v : A \Leftrightarrow I \Vdash_1 u = v : A$

2. *If* $I \Vdash_0 A = B$, *then* $I \Vdash_1 A = B$.

Proof. By simultaneous induction on $I \Vdash_0 A$ and $I \Vdash_0 A = B$. □

We will write $I \Vdash A$ if there is a derivation of $I \Vdash_\ell A$ for some ℓ; etc. Such derivations will be ordered lexicographically, i.e., $I \Vdash_0 A$ derivations are ordered before $I \Vdash_1 A$ derivations.

Lemma 7.3.5.

1. $I \Vdash_\ell A \Rightarrow I \Vdash_\ell A = A$
2. $I \Vdash_\ell A \;\&\; I \Vdash_\ell u : A \Rightarrow I \Vdash_\ell u = u : A$

Proof. Simultaneously, by induction on ℓ and side induction on $I \Vdash_\ell A$. In the case GL-C, to see (2), note that from the assumption $I \Vdash u : B$ with B being $\mathsf{Glue}\,[\varphi \mapsto (T, w)]\,A$ we get in particular

$$I, \varphi \Vdash w = w : \mathsf{Equiv}\,T\,A \Rightarrow I \Vdash \mathsf{unglue}\,[\varphi \mapsto w]\,u = \mathsf{unglue}\,[\varphi \mapsto w]\,u : A.$$

But by IH, the premise follows from $I, \varphi \Vdash w : \mathsf{Equiv}\,T\,A$; moreover, $I, \varphi \Vdash u = u : B$ is immediate by IH, showing $I \Vdash u = u : B$. □

Lemma 7.3.6 (Monotonicity/Substitution)**.** *For* $f \colon J \to I$ *we have*

1. $I \Vdash_\ell A \Rightarrow J \Vdash_\ell Af$,
2. $I \Vdash_\ell A = B \Rightarrow J \Vdash_\ell Af = Bf$,
3. $I \Vdash_\ell A \;\&\; I \Vdash_\ell u : A \Rightarrow J \Vdash_\ell uf : Af$,
4. $I \Vdash_\ell A \;\&\; I \Vdash_\ell u = v : A \Rightarrow J \Vdash_\ell uf = vf : Af$.

Moreover, the respective heights of the derivations don't increase.

Proof. By induction on ℓ and side induction on $I \Vdash_\ell A$ and $I \Vdash_\ell A = B$. The definition of computability predicates and relations is lead such that this proof is immediate. For instance, note for (1) in the case GL-C, i.e.,

$$\frac{1 \neq \varphi \in \mathbb{F}(I) \qquad I, 1 \Vdash_\ell A \qquad I, \varphi \Vdash_\ell \mathsf{Equiv}\,T\,A \qquad I, \varphi \Vdash_\ell w : \mathsf{Equiv}\,T\,A \qquad I, \varphi \Vdash_\ell \mathsf{Glue}\,[\varphi \mapsto (T, w)]\,A}{I \Vdash_\ell \mathsf{Glue}\,[\varphi \mapsto (T, w)]\,A}\ \textsc{Gl-C}$$

we distinguish cases: if $\varphi f = 1$, then $J \Vdash_\ell \mathsf{Glue}\,[\varphi f \mapsto (Tf, wf)]\,Af$ by the premise $I, \varphi \Vdash_\ell \mathsf{Glue}\,[\varphi \mapsto (T, w)]\,A$; in case $\varphi f \neq 1$ we can use the same rule again. □

Lemma 7.3.7.

1. $I \Vdash A \Rightarrow I \Vdash A{\downarrow}$
2. $I \Vdash A = B \Rightarrow I \Vdash A{\downarrow} = B{\downarrow}$

3. $I \Vdash A \;\&\; I \Vdash u : A \Rightarrow I \Vdash u : A{\downarrow}$
4. $I \Vdash A \;\&\; I \Vdash u = v : A \Rightarrow I \Vdash u = v : A{\downarrow}$
5. $I \Vdash u : \mathsf{N} \Rightarrow I \Vdash u{\downarrow} : \mathsf{N}$
6. $I \Vdash u = v : \mathsf{N} \Rightarrow I \Vdash u{\downarrow} = v{\downarrow} : \mathsf{N}$

Moreover, the respective heights of the derivations don't increase.

Proof. (1) By induction on $I \Vdash A$. All cases were A is an introduction are immediate since then $A{\downarrow}$ is A. It only remains the case Ni-C:

$$\frac{A \text{ n.i.} \qquad \forall f\colon J \to I(Af! \;\&\; J \Vdash Af{\downarrow}) \qquad \forall f\colon J \to I \forall g\colon K \to J(K \Vdash Af{\downarrow}g = Afg{\downarrow})}{I \Vdash A}\ \text{Ni-C}$$

We have $I \Vdash A{\downarrow}$ as this is one of the premises.

(5) By induction on $I \Vdash u : \mathsf{N}$ similarly to the last paragraph.

(2) By induction on $I \Vdash A = B$. The only case where a reduct may happen is Ni-E, in which $I \Vdash A{\downarrow} = B{\downarrow}$ is a premise. Similar for (6).

(3) and (4): By induction on $I \Vdash A$, where the only interesting case is Ni-C, in which what we have to show holds by definition. □

Lemma 7.3.8.

1. *If* $I \Vdash A = B$, *then*
   (a) $I \Vdash u : A \Leftrightarrow I \Vdash u : B$, *and*
   (b) $I \Vdash u = v : A \Leftrightarrow I \Vdash u = v : B$.
2. $I \Vdash A = B \;\&\; I \Vdash B = C \Rightarrow I \Vdash A = C$
3. *Given* $I \Vdash A$ *we get*
$$I \Vdash u = v : A \;\&\; I \Vdash v = w : A \Rightarrow I \Vdash u = w : A.$$
4. $I \Vdash A = B \Rightarrow I \Vdash B = A$
5. $I \Vdash A \;\&\; I \Vdash u = v : A \Rightarrow I \Vdash v = u : A$

Proof. We prove the statement for "$\Vdash_\ell$" instead of "$\Vdash$" by main induction on ℓ (i.e., we prove the statement for "$\Vdash_0$" before the statement for "$\Vdash_1$"); the statement for "$\Vdash$" follows then from Lemma 7.3.4.

Simultaneously by threefold induction on $I \Vdash_\ell A$, $I \Vdash_\ell B$, and $I \Vdash_\ell C$. (Alternatively by induction on the (natural) sum of the heights of $I \Vdash_\ell A$, $I \Vdash_\ell B$, and $I \Vdash_\ell C$; we only need to be able to apply the IH if the complexity of at least one derivation decreases and the others won't increase.) In the proof below we will omit ℓ to simplify notation, except in cases where the level matters.

(1) By distinguishing cases on $I \Vdash A = B$. We only give the argument for (1a) as (1b) is very similar except in case Gl-E. The cases N-E and U-E are trivial.

Case Pi-E. Let $I \Vdash w : (x : A) \to B$ and we show $I \Vdash w : (x : A') \to B'$. For $f \colon J \to I$ let $J \Vdash u : A'f$; then by IH (since $J \Vdash Af = A'f$) we get $J \Vdash u : Af$, and thus $J \Vdash wf\,u : B(f, x/u)$; again by IH we obtain $J \Vdash wf\,u : B'(f, x/u)$. Now assume $J \Vdash u = v : A'f$; so by IH, $J \Vdash u = v : Af$, and thus $J \Vdash wf\,u = wf\,v : B(f, x/u)$. Again by IH, we conclude $J \Vdash wf\,u = wf\,v : B'(f, x/u)$. Thus we have proved $I \Vdash w : (x : A') \to B'$.

Case Si-E. For $I \Vdash w : (x : A) \times B$ and we show $I \Vdash w : (x : A') \times B'$. We have $I \Vdash w.1 : A$ and $I \Vdash w.2 : B(x/w.1)$. So by IH, $I \Vdash w.1 : A'$; moreover, we have $I \Vdash B(x/w.1) = B'(x/w.1)$; so, again by IH, we conclude with $I \Vdash w.2 : B'(x/w.1)$.

Case Pa-E. Let $I \Vdash u : \mathsf{Path}\, A\, a_0\, a_1$ and we show $I \Vdash u : \mathsf{Path}\, B\, b_0\, b_1$. Given $f \colon J \to I$ and $r \in \mathbb{F}(J)$ we have $J \Vdash uf\,r : Af$ and thus $J \Vdash uf\,r : Bf$ by IH. We have to check that the endpoints match: $I \Vdash u\,0 = a_0 : A$ by assumption; moreover, $I \Vdash a_0 = b_0 : A$, so by IH (3), $I \Vdash u\,0 = b_0 : A$, thus again using the IH, $I \Vdash u\,0 = b_0 : B$.

Case Gl-E. Let us abbreviate $\mathsf{Glue}\,[\varphi \mapsto (T, w)]\,A$ by D, and $\mathsf{Glue}\,[\varphi \mapsto (T', w')]\,A'$ by D'.

(1a) Let $I \Vdash u : D$, i.e., $I, \varphi \Vdash u : D$ and

$$J \Vdash \mathsf{unglue}\,[\varphi f \mapsto w'']\,uf = \mathsf{unglue}\,[\varphi f \mapsto wf]\,uf : Af \tag{7.1}$$

whenever $f \colon J \to I$ and $J, \varphi f \Vdash w'' = wf : \mathsf{Equiv}\,Tf\,Af$. Directly by IH we obtain $I, \varphi \Vdash u : D'$. Now let $f \colon J \to I$ and $J, \varphi f \Vdash w'' = w'f : \mathsf{Equiv}\,T'f\,A'f$; by IH, also $J, \varphi f \Vdash w'' = w'f : \mathsf{Equiv}\,Tf\,Af$. Moreover, we have $J, \varphi f \Vdash wf = w'f : \mathsf{Equiv}\,Tf\,Af$, hence (7.1) gives (together with symmetry and transitivity, applicable by IH)

$$J \Vdash \mathsf{unglue}\,[\varphi f \mapsto w'']\,uf = \mathsf{unglue}\,[\varphi f \mapsto wf]\,uf : Af, \text{ and}$$
$$J \Vdash \mathsf{unglue}\,[\varphi f \mapsto w'f]\,uf = \mathsf{unglue}\,[\varphi f \mapsto wf]\,uf : Af.$$

Hence, transitivity and symmetry (which we can apply by IH) give that the above left-hand sides are forced equal of type Af, applying the IH (1b) gives that they are forced equal of type $A'f$, and thus $I \Vdash u : D'$.

(1b) Let $I \Vdash u = v : D$, so we have $I, \varphi \Vdash u = v : D$ and

$$I \Vdash \mathsf{unglue}\,[\varphi \mapsto w]\,u = \mathsf{unglue}\,[\varphi \mapsto w]\,v : A \tag{7.2}$$

By IH, we get $I, \varphi \Vdash u = v : D'$ from $I, \varphi \Vdash u = v : D$. Note that we also have $I \Vdash u : D$ and $I \Vdash v : D$, and thus

$$I \Vdash \mathsf{unglue}\,[\varphi \mapsto w]\,u = \mathsf{unglue}\,[\varphi \mapsto w']\,u : A, \text{ and}$$
$$I \Vdash \mathsf{unglue}\,[\varphi \mapsto w]\,v = \mathsf{unglue}\,[\varphi \mapsto w']\,v : A$$

and thus with (7.2) and transitivity and symmetry (which we can apply by IH) we obtain $I \Vdash \mathsf{unglue}\,[\varphi \mapsto w']\,u = \mathsf{unglue}\,[\varphi \mapsto w']\,v : A$, hence also at type A' by IH. Therefore we proved $I \Vdash u = v : D'$.

Case NI-E. Let $I \Vdash u : A$; we have to show $I \Vdash u : B$.

Subcase B is non-introduced. Then we have to show $J \Vdash uf : Bf{\downarrow}$ for $f\colon J \to I$. We have $J \Vdash Af{\downarrow} = Bf{\downarrow}$ and since $I \Vdash B$ is non-introduced, the derivation $J \Vdash Bf{\downarrow}$ is shorter than $I \Vdash B$, and the derivation $J \Vdash Af{\downarrow}$ is not higher than $I \Vdash A$ by Lemma 7.3.7. Moreover, also $J \Vdash uf : Af$ so by Lemma 7.3.7 (3) we get $J \Vdash uf : Af{\downarrow}$, and hence by IH, $J \Vdash uf : Bf{\downarrow}$.

Subcase B is introduced. We have $I \Vdash u : A{\downarrow}$ and $I \Vdash A{\downarrow} = B{\downarrow}$ but $B{\downarrow}$ is B, and $I \Vdash A{\downarrow}$ has a shorter derivation than $I \Vdash A$, so $I \Vdash u : B$ by IH.

(2) Let us first handle the cases where A, B, or C is non-introduced. It is enough to show $J \Vdash Af{\downarrow} = Cf{\downarrow}$ (if A and C are both introduced, this entails $I \Vdash A = C$ for f the identity). We have $J \Vdash Af{\downarrow} = Bf{\downarrow}$ and $J \Vdash Bf{\downarrow} = Cf{\downarrow}$. None of the respective derivations get higher (by Lemma 7.3.7) but one gets shorter since one of the types is non-introduced. Thus the claim follows by IH.

It remains to look at the cases where all are introduced; in this case both equalities have to be derived by the same rule. We distinguish cases on the rule.

Case N-E. Trivial. *Case* SI-E. Similar to PI-E below. *Case* PA-E and GL-E. Use the IH. *Case* U-E. Trivial.

Case PI-E. Let use write A as $(x : A') \to A''$ and similar for B and C. We have $I \Vdash A' = B'$ and $I \Vdash B' = C'$, and so by IH, we get $I \Vdash A' = C'$; for $J \Vdash u : A'f$ where $f\colon J \to I$ it remains to be shown that $J \Vdash A''(f, x/u) = C''(f, x/u)$. By IH, we also have $J \Vdash u : B'f$, so we have

$$J \Vdash A''(f, x/u) = B''(f, x/u) \text{ and } J \Vdash B''(f, x/u) = C''(f, x/u)$$

and can conclude by the IH.

(3) By cases on $I \Vdash A$. All cases follow immediately using the IH, except for N-C and U-C. In case N-C, we show transitivity by a side induction on the (natural) sum of the height of the derivations $I \Vdash u = v : \mathsf{N}$ and $I \Vdash v = w : \mathsf{N}$. If one of u,v, or w is non-introduced, we get that one of the derivations $J \Vdash uf{\downarrow} = vf{\downarrow} : \mathsf{N}$ and $J \Vdash vf{\downarrow} = wf{\downarrow} : \mathsf{N}$ is shorter (and the other doesn't get higher), so by SIH, $J \Vdash uf{\downarrow} = wf{\downarrow} : \mathsf{N}$ which entails $I \Vdash u = w : \mathsf{N}$. Otherwise, $I \Vdash u = v : \mathsf{N}$ and $I \Vdash v = w : \mathsf{N}$ have to be derived with the same rule and $I \Vdash u = w : \mathsf{N}$ easily follows (using the SIH in the successor case).

In case U-C, we have $I \Vdash_1 u = v : \mathsf{U}$ and $I \Vdash_1 v = w : \mathsf{U}$, i.e., $I \Vdash_0 u = v$ and $I \Vdash_0 v = w$. We want to show $I \Vdash_1 u = w : \mathsf{U}$, i.e., $I \Vdash_0 u = w$. But by IH(ℓ), we can already assume the lemma is proven for $\ell = 0$, hence can use transitivity and deduce $I \Vdash_0 u = w$.

The proofs of (4) and (5) are by distinguishing cases and are straightforward. □

Remark 7.3.9. Now that we have established transitivity, proving computability for Π-types can also achieved as follows. Given we have $I \Vdash (x : A) \to B$ and derivations $I \vdash w : (x : A) \to B$, $I \vdash w' : (x : A) \to B$, and $I \vdash w = w' : (x : A) \to B$, then $I \Vdash w = w' : (x : A) \to B$ whenever we have

$$\forall f\colon J \to I \forall u, v (J \Vdash u = v : Af \Rightarrow J \Vdash wf\, u = w'f\, v : B(f, x/u)).$$

(In particular, this gives $I \Vdash w : (x : A) \to B$ and $I \Vdash w' : (x : A) \to B$.)

Likewise, given $I \vdash (x : A) \to B$, $I \vdash (x : A') \to B'$, and $I \vdash (x : A) \to B = (x : A') \to B'$, we get $I \Vdash (x : A) \to B = (x : A') \to B'$ whenever $I \Vdash A = A'$ and

$$\forall f \colon J \to I \forall u, v (J \Vdash u = v : Af \Rightarrow J \Vdash B(f, x/u) = B'(f, x/v)).$$

Lemma 7.3.10.

1. $I \Vdash A \Rightarrow I \Vdash A = A{\downarrow}$
2. $I \Vdash u : \mathsf{N} \Rightarrow I \Vdash u = u{\downarrow} : \mathsf{N}$

Proof. (1) We already proved $I \Vdash A{\downarrow}$ in Lemma 7.3.7 (1). By induction on $I \Vdash A$. All cases where A is an introduction are immediate since then $A{\downarrow}$ is A. It only remains the case Ni-C:

$$\frac{A \text{ n.i.} \qquad \forall f \colon J \to I (Af! \ \ \& \ \ J \Vdash Af{\downarrow}) \qquad \forall f \colon J \to I \forall g \colon K \to J (K \Vdash Af{\downarrow}g = Afg{\downarrow})}{I \Vdash A} \text{ Ni-C}$$

We now show $I \Vdash A = A{\downarrow}$; since A is non-introduced we have to show $J \Vdash Af{\downarrow} = (A{\downarrow}f){\downarrow}$ for $f \colon J \to I$. $I \Vdash A{\downarrow}$ has a shorter derivation than $I \Vdash A$, thus so has $J \Vdash A{\downarrow}f$; hence by IH, $J \Vdash A{\downarrow}f = (A{\downarrow}f){\downarrow}$. We also have $J \Vdash A{\downarrow}f = Af{\downarrow}$ by definition of $I \Vdash A$, and thus we obtain $J \Vdash Af{\downarrow} = (A{\downarrow}f){\downarrow}$ using symmetry and transitivity.

(2) Similar, by induction on $I \Vdash u : \mathsf{N}$. □

Lemma 7.3.11 (Expansion Lemma)**.** *Let $I \Vdash_\ell A$ and $I \vdash u : A$; then:*

$$\frac{\forall f \colon J \to I (uf!^{Af} \ \ \& \ \ J \Vdash_\ell uf{\downarrow}^{Af} : Af) \qquad \forall f \colon J \to I (J \Vdash_\ell uf{\downarrow} = u{\downarrow}f : Af)}{I \Vdash_\ell u : A \ \ \& \ \ I \Vdash_\ell u = u{\downarrow} : A}$$

In particular, if $I \vdash u \succ_{\mathsf{s}} v : A$ and $I \Vdash_\ell v : A$, then $I \Vdash_\ell u : A$ and $I \Vdash_\ell u = v : A$.

Proof. By induction on $I \Vdash A$. We will omit the level annotation ℓ whenever it is inessential.

Case N-C. We have to show $K \Vdash uf{\downarrow}g = ufg{\downarrow} : \mathsf{N}$ for $f \colon J \to I$ and $g \colon K \to J$; we have $J \Vdash uf{\downarrow} = u{\downarrow}f : \mathsf{N}$, thus $K \Vdash uf{\downarrow}g = u{\downarrow}fg : \mathsf{N}$. Moreover, $K \Vdash u{\downarrow}fg = ufg{\downarrow} : \mathsf{N}$ by assumption, and thus by transitivity $K \Vdash uf{\downarrow}g = u{\downarrow}fg = ufg{\downarrow} : \mathsf{N}$. (Likewise one shows that the data in the premise of the lemma is closed under substitution.)

$I \Vdash u = u{\downarrow} : \mathsf{N}$ holds by Lemma 7.3.7 (2).

Case Pi-C. First, let $J \Vdash a : Af$ for $f \colon J \to I$. We have

$$K \vdash (uf\,a)g \succ (ufg){\downarrow}(ag) : B(fg, x/ag)$$

for $g\colon K \to J$, and also $K \Vdash (uf g)\!\downarrow (ag) : B(fg, x/ag)$ and we have the compatibility condition

$$\begin{aligned} K \Vdash (uf\,a)g\!\downarrow &= ((ufg)\!\downarrow)\,(ag) = (uf\!\downarrow g)\,(ag) \\ &= (uf\!\downarrow a)g = (uf\,a)\!\downarrow g : B(fg, x/ag), \end{aligned}$$

so by IH, $J \Vdash uf\,a : B(f, x/a)$ and $J \Vdash uf\,a = uf\!\downarrow a : B(f, x/a)$. Since also $J \Vdash uf\!\downarrow = u\!\downarrow f : ((x : A) \to B)f$ we also get $J \Vdash uf\,a = u\!\downarrow f\,a : B(f, x/a)$.

Now if $J \Vdash a = b : Af$, we also have $J \Vdash a : Af$ and $J \Vdash b : Af$, so like above we get $J \Vdash uf\,a = uf\!\downarrow a : B(f, x/a)$ and $J \Vdash uf\,b = uf\!\downarrow b : B(f, x/b)$ (and thus also $J \Vdash uf\,b = uf\!\downarrow b : B(f, x/a)$). Moreover, $J \Vdash uf\!\downarrow a = uf\!\downarrow b : B(f, x/a)$ and hence we can conclude $J \Vdash uf\,a = uf\,b : B(f, x/a)$ by transitivity and symmetry. Thus we showed both $I \Vdash u : (x : A) \to B$ and $I \Vdash u = u\!\downarrow : (x : A) \to B$.

Case Si-C. Clearly we have $(u.1f)\!\downarrow = (uf\!\downarrow).1$, $J \Vdash (uf\!\downarrow).1 : Af$, and

$$J \Vdash (u.1f)\!\downarrow = (uf\!\downarrow).1 = (u\!\downarrow f).1 = (u\!\downarrow.1)f = (u.1)\!\downarrow f : Af$$

so the IH gives $I \Vdash u.1 : A$ and $I \Vdash u.1 = (u\!\downarrow).1 : A$. Likewise $(u.2f)\!\downarrow = (uf\!\downarrow).2$ and $J \Vdash (uf\!\downarrow).2 : B(f, x/uf\!\downarrow.1)$, hence we also get $J \Vdash (uf\!\downarrow).2 : B(f, x/uf.1)$; as above one shows $J \Vdash (u.2f)\!\downarrow = u.2\!\downarrow f : B(f, x/uf.1)$, applying the IH once more to obtain $I \Vdash u.2 = u\!\downarrow.2 : B(x/u.1))$ which was what remained to be proven.

Case Pa-C. Let us write $\mathsf{Path}\,A\,v\,w$ for the type and let $f\colon J \to I$, $r \in \mathbb{I}(J)$, and $g\colon K \to J$. We have

$$K \vdash (uf\,r)g \succ (ufg)\!\downarrow (rg) : Afg$$

and $K \Vdash (ufg)\!\downarrow (rg) : Afg$; moreover,

$$K \Vdash (uf\,r)g\!\downarrow = (ufg)\!\downarrow (rg) = (uf\!\downarrow g)\,(rg) = (uf\!\downarrow r)g = (uf\,r)\!\downarrow g : Afg.$$

Thus by IH, $J \Vdash uf\,r : Af$ and

$$J \Vdash uf\,r = uf\!\downarrow r = u\!\downarrow f\,r : Af. \tag{7.3}$$

So we obtain $I \Vdash u\,0 = u\!\downarrow 0 = v : A$ and $I \Vdash u\,1 = u\!\downarrow 1 = w : A$, and hence $I \Vdash u : \mathsf{Path}\,A\,v\,w$; $I \Vdash u = u\!\downarrow : \mathsf{Path}\,A\,v\,w$ follows from (7.3).

Case Gl-C. Abbreviate $\mathsf{Glue}\,[\varphi \mapsto (T, w)]\,A$ by B. Note that we have $\varphi \neq 1$. First, we claim that for any $f\colon J \to I$, $J \Vdash b : Bf$, and $J, \varphi f \Vdash w' = wf : \mathsf{Equiv}\,Tf\,Af$,

$$J, \varphi f \Vdash \mathsf{unglue}\,[\varphi f \mapsto w']\,b = w'.1\,b : Af. \tag{7.4}$$

(In particular both sides are computable.) Indeed, for $g\colon K \to J$ with $\varphi fg = 1$ we have that

$$K \vdash (\mathsf{unglue}\,[\varphi f \mapsto w']\,b)g \succ_{\mathsf{s}} w'g.1\,(bg) : Afg$$

and $K \Vdash w'g.1\,(bg) : Afg$ since $I, \varphi \Vdash B = T$ (which in turn follows from Lemma 7.3.10 (1)). Thus by IH ($J \Vdash Af$ has a shorter derivation than $I \Vdash B$),

$$K \Vdash (\mathsf{unglue}\,[\varphi f \mapsto w']\,b)g = (w'.1\,b)g : Afg$$

as claimed.

Next, let $f\colon J \to I$ such that $\varphi f = 1$; then using the IH ($J \Vdash Bf$ has a shorter derivation than $I \Vdash B$), we get $J \Vdash uf : Bf$ and $J \Vdash uf = uf\!\downarrow : Bf$, and hence also $J \Vdash uf = u\!\downarrow f : Bf$ (since $J \Vdash uf\!\downarrow = u\!\downarrow f : Bf$). That is, we proved

$$I, \varphi \Vdash u : B \text{ and } I, \varphi \Vdash u = u\!\downarrow : B. \tag{7.5}$$

We will now first show

$$J \Vdash \mathsf{unglue}\,[\varphi f \mapsto w']\,uf = (\mathsf{unglue}\,[\varphi f \mapsto w']\,uf)\!\downarrow : Af \tag{7.6}$$

for and $f\colon J \to I$ and $J, \varphi f \Vdash w' = wf : \mathsf{Equiv}\,Tf\,Af$. We can assume that w.l.o.g. $\varphi f \neq 1$, since if $\varphi f = 1$, $J \Vdash uf : Bf$ by (7.5), and (7.6) follows from (7.4) noting that its right-hand side is the reduct. We will use the IH to show (7.6), so let us analyze the reduct:

$$(\mathsf{unglue}\,[\varphi f \mapsto w']\,uf)g\!\downarrow = \begin{cases} (w'g.1)\,(ufg) & \text{if } \varphi fg = 1, \\ \mathsf{unglue}\,[\varphi f \mapsto w'g]\,(ufg\!\downarrow) & \text{otherwise.} \end{cases} \tag{7.7}$$

where $g\colon K \to J$. In either case, the reduct is computable: in the first case, use (7.5) and $J \Vdash w'.1 : T \to A$ together with the observation $I, \varphi \Vdash B = T$; in the second case this follows from $J \Vdash ufg\!\downarrow : Bfg$. In order to apply the IH, it remains to verify

$$K \Vdash (\mathsf{unglue}\,[\varphi f \mapsto w']\,uf)g\!\downarrow = (\mathsf{unglue}\,[\varphi f \mapsto w']\,uf)\!\downarrow g : Afg.$$

In case $\varphi fg \neq 1$, we have

$$\begin{aligned} K \Vdash\ & \mathsf{unglue}\,[\varphi fg \mapsto w'g]\,(ufg\!\downarrow) \\ &= \mathsf{unglue}\,[\varphi fg \mapsto wfg]\,(ufg\!\downarrow) && \text{since } K \Vdash ufg\!\downarrow : Bfg \\ &= \mathsf{unglue}\,[\varphi fg \mapsto wfg]\,(uf\!\downarrow g) && \text{since } K \Vdash ufg\!\downarrow = uf\!\downarrow g : Bfg \\ &= \mathsf{unglue}\,[\varphi fg \mapsto w'g]\,(uf\!\downarrow g) : Afg && \text{since } K \Vdash uf\!\downarrow g : Bfg \end{aligned}$$

which is what we had to show in this case. In case $\varphi fg = 1$, we have to prove

$$K \Vdash (w'g.1)\,(ufg) = \mathsf{unglue}\,[\varphi f \mapsto w'g]\,(uf\!\downarrow g) : Afg. \tag{7.8}$$

But by (7.5) we have $K \Vdash ufg = ufg\!\downarrow = uf\!\downarrow g : Bfg$, so also $K \Vdash (w'g.1)\,(ufg) = (w'g.1)\,(uf\!\downarrow g) : Afg$, so (7.8) follows from (7.4) using $J \Vdash uf\!\downarrow : Bf$. This concludes the proof of (7.6).

As w' could have been wf we also get

$$J \Vdash \mathsf{unglue}\,[\varphi f \mapsto wf]\,uf = (\mathsf{unglue}\,[\varphi f \mapsto wf]\,uf)\!\downarrow : Af. \tag{7.9}$$

In order to prove $I \Vdash u : B$ it remains to check that the left-hand side of (7.6) is forced equal to the left-hand side of (7.9); so we can simply check this for the respective right-hand sides: in case $\varphi f = 1$, these are $w'.1\,uf$ and $wf.1\,uf$, respectively, and hence forced equal since $J \Vdash w' = wf : \mathsf{Equiv}\,Tf\,Af$; in case $\varphi f \neq 1$, we have to show

$$J \Vdash \mathsf{unglue}\,[\varphi f \mapsto w']\,(uf{\downarrow}) = \mathsf{unglue}\,[\varphi f \mapsto wf]\,(uf{\downarrow}) : Af$$

which simply follows since $J \Vdash uf{\downarrow} : Bf$.

In order to prove $I \Vdash u = u{\downarrow} : B$ it remains to check

$$I \Vdash \mathsf{unglue}\,[\varphi \mapsto w]\,u = \mathsf{unglue}\,[\varphi \mapsto w]\,(u{\downarrow}) : A,$$

but this is (7.9) in the special case where f is the identity.

Case U-C. Let us write B for u. We have to prove $I \Vdash_1 B : \mathsf{U}$ and $I \Vdash_1 B = B{\downarrow} : \mathsf{U}$, i.e., $I \Vdash_0 B$ and $I \Vdash_0 B = B{\downarrow}$. By Lemma 7.3.10 (1), it suffices to prove the former. For $f\colon J \to I$ we have

$$J \vdash Bf \succ Bf{\downarrow}^{\mathsf{U}} : \mathsf{U}$$

and hence also

$$J \vdash Bf \succ Bf{\downarrow}^{\mathsf{U}}$$

i.e., $Bf!$, and $Bf{\downarrow}$ is $Bf{\downarrow}^{U}$; since $J \Vdash_1 Bf{\downarrow} : \mathsf{U}$ we have $J \Vdash_0 Bf{\downarrow}$, and likewise $J \Vdash_0 Bf{\downarrow} = B{\downarrow}f$. Moreover, if also $g\colon K \to J$, we obtain $K \Vdash_0 Bfg{\downarrow} = B{\downarrow}fg$ from the assumption. Hence $K \Vdash_0 Bf{\downarrow}g = B{\downarrow}fg = Bfg{\downarrow}$, therefore $I \Vdash_0 B$ what we had to show.

Case Ni-C. Then $I \Vdash A{\downarrow}$ has a shorter derivation than $I \Vdash A$; moreover, for $f\colon J \to I$ we have $J \vdash uf \succ uf{\downarrow}^{Af} : Af$ so also $J \vdash uf \succ uf{\downarrow}^{Af} : A{\downarrow}f$ since $J \vdash Af = A{\downarrow}f$. By Lemma 7.3.7 (1), $I \Vdash A = A{\downarrow}$ so also $J \Vdash uf{\downarrow} : A{\downarrow}f$ and $J \Vdash uf{\downarrow} = u{\downarrow}f : A{\downarrow}f$, and hence by IH, $I \Vdash u : A{\downarrow}$ and $I \Vdash u = u{\downarrow} : A{\downarrow}$, so also $I \Vdash u : A$ and $I \Vdash u = u{\downarrow} : A$ using $I \Vdash A = A{\downarrow}$ again. □

7.4 Soundness

The aim of this section is to prove canonicity as stated in the introduction. We will do so by showing that each computable instance of a judgment derived in cubical type theory is computable (allowing free name variables)—this is the content of the Soundness Theorem below.

We first extend the computability predicates to contexts and substitutions. $\boxed{\Vdash \Gamma}$ assuming $\Gamma \vdash$.

$$\frac{}{\Vdash \diamond} \qquad \frac{\Vdash \Gamma \qquad i \notin \mathrm{dom}(\Gamma)}{\Vdash \Gamma, i : \mathbb{I}} \qquad \frac{\Vdash \Gamma \qquad \Gamma \vdash \varphi : \mathbb{F}}{\Vdash \Gamma, \varphi}$$

$$\frac{\Vdash \Gamma \qquad \forall I \forall \sigma (I \Vdash \sigma : \Gamma \Rightarrow I \Vdash A\sigma) \qquad \forall I \forall \sigma, \tau (I \Vdash \sigma = \tau : \Gamma \Rightarrow I \Vdash A\sigma = A\tau) \qquad x \notin \mathrm{dom}(\Gamma)}{\Vdash \Gamma, x : A}$$

$\boxed{I \Vdash \sigma : \Gamma}$ by induction on $\Vdash \Gamma$ assuming $I \vdash \sigma : \Gamma$.

$$\frac{}{I \Vdash () : \diamond} \qquad \frac{I \Vdash \sigma : \Gamma \qquad r \in \mathbb{I}(I)}{I \Vdash (\sigma, i/r) : \Gamma, i : \mathbb{I}} \qquad \frac{I \Vdash \sigma : \Gamma \qquad \varphi\sigma = 1}{I \Vdash \sigma : \Gamma, \varphi}$$

$$\frac{I \Vdash \sigma : \Gamma \qquad I \Vdash u : A\sigma}{I \Vdash (\sigma, x/u) : \Gamma, x : A}$$

$\boxed{I \Vdash \sigma = \tau : \Gamma}$ by induction on $\Vdash \Gamma$, assuming $I \Vdash \sigma : \Gamma$, $I \Vdash \tau : \Gamma$, and $I \vdash \sigma = \tau : \Gamma$.

$$\frac{}{I \Vdash () = () : \diamond} \qquad \frac{I \Vdash \sigma = \tau : \Gamma \qquad r \in \mathbb{I}(I)}{I \Vdash (\sigma, i/r) = (\tau, i/r) : \Gamma, i : \mathbb{I}}$$

$$\frac{I \Vdash \sigma = \tau : \Gamma \qquad \varphi\sigma = \varphi\tau = 1}{I \Vdash \sigma = \tau : \Gamma, \varphi} \qquad \frac{I \Vdash \sigma = \tau : \Gamma \qquad I \Vdash u = v : A\sigma}{I \Vdash (\sigma, x/u) = (\tau, x/v) : \Gamma, x : A}$$

We write $I \Vdash r : \mathbb{I}$ for $r \in \mathbb{I}(I)$, $I \Vdash r = s : \mathbb{I}$ for $r = s \in \mathbb{I}(I)$, and likewise $I \Vdash \varphi : \mathbb{F}$ for $\varphi \in \mathbb{F}(I)$, $I \Vdash \varphi = \psi : \mathbb{F}$ for $\varphi = \psi \in \mathbb{F}(I)$. In the next definition we allow A to be $\mathbb{F}$ or $\mathbb{I}$, and also correspondingly for a and b to range over interval and face lattice elements.

Definition 7.4.1.

$$\begin{array}{lll}
\Gamma \models & :\Leftrightarrow & \Vdash \Gamma \\
\Gamma \models A = B & :\Leftrightarrow & \Gamma \vdash A = B \ \ \& \ \ \Gamma \models \ \& \\
 & & \forall I, \sigma, \tau (I \Vdash \sigma = \tau : \Gamma \Rightarrow I \Vdash A\sigma = B\tau) \\
\Gamma \models A & :\Leftrightarrow & \Gamma \vdash A \ \ \& \ \ \Gamma \models A = A \\
\Gamma \models a = b : A & :\Leftrightarrow & \Gamma \vdash a = b : A \ \ \& \ \ \Gamma \models A \ \ \& \\
 & & \forall I, \sigma, \tau (I \Vdash \sigma = \tau : \Gamma \Rightarrow I \Vdash a\sigma = b\tau : A\sigma) \\
\Gamma \models a : A & :\Leftrightarrow & \Gamma \vdash a : A \ \ \& \ \ \Gamma \models a = a : A \\
\Gamma \models \sigma = \tau : \Delta & :\Leftrightarrow & \Gamma \vdash \sigma = \tau : \Delta \ \ \& \ \ \Gamma \models \ \& \ \ \Delta \models \ \& \\
 & & \forall I, \delta, \gamma (I \Vdash \delta = \gamma : \Gamma \Rightarrow I \Vdash \sigma\delta = \tau\gamma : \Delta) \\
\Gamma \models \sigma : \Delta & :\Leftrightarrow & \Gamma \vdash \sigma : \Delta \ \ \& \ \ \Gamma \models \sigma = \sigma : \Delta
\end{array}$$

Remark 7.4.2.

1. For each I we have $\Vdash I$, and $J \Vdash \sigma : I$ iff $\sigma \colon J \to I$; likewise, $J \Vdash \sigma = \tau : I$ iff $\sigma = \tau$.
2. For computability of contexts and substitutions monotonicity and partial equivalence properties hold analogous to computability of types and terms.
3. Given $\Vdash \Gamma$ and $I \Vdash \sigma = \tau : \Gamma$, then for any $\Gamma \vdash \varphi : \mathbb{F}$ we get $\varphi\sigma = \varphi\tau \in \mathbb{F}(I)$ since $\varphi\sigma$ and $\varphi\tau$ only depend on the name assignments of σ and τ which have to agree by $I \Vdash \sigma = \tau : \Gamma$. Similarly for $\Gamma \vdash r : \mathbb{I}$.

4. The definition of "$\models$" slightly deviates from the approach we had in the definition of "$\Vdash$" as, say, $\Gamma \models A$ is defined in terms of $\Gamma \models A = A$. Note that by the properties we already established about "$\Vdash$" we get that $\Gamma \models A = B$ implies $\Gamma \models A$ and $\Gamma \models B$ (given we know $\Gamma \vdash A$ and $\Gamma \vdash B$, respectively); and, likewise, $\Gamma \models a = b : A$ entails $\Gamma \models a : A$ and $\Gamma \models b : A$ (given $\Gamma \vdash a : A$ and $\Gamma \vdash b : A$, respectively). Also, note that in the definition of, say, $\Gamma \models A$, the condition

$$\forall I, \sigma, \tau (I \Vdash \sigma = \tau : \Gamma \Rightarrow I \Vdash A\sigma = A\tau)$$

implies

$$\forall I, \sigma (I \Vdash \sigma : \Gamma \Rightarrow I \Vdash A\sigma).$$

In fact, we will often have to establish the latter condition first when showing the former.

5. $I \models A = B$ iff $I \Vdash A = B$, and $I \models a = b : A$ iff $I \Vdash A$ and $I \Vdash a = b : A$; moreover, given $I \models A$ and $I, x : A \vdash B$, then $I, x : A \models B$ iff

$$\forall f \colon J \to I \forall u (J \Vdash u : Af \Rightarrow J \Vdash B(f, x/u)) \;\&$$
$$\forall f \colon J \to I \forall u, v (J \Vdash u = v : Af \Rightarrow J \Vdash B(f, x/u) = B(f, x/v))$$

(Note that the second formula in the above display implies the first.) Thus the premises of Pi-C and Si-C are simply $I \models A$ and $I, x : A \models B$. Also, $I, \varphi \Vdash A = B$ iff $I, \varphi \models A = B$; and $I, \varphi \models a = b : A$ iff $I, \varphi \Vdash A$ and $I, \varphi \Vdash a = b : A$.

6. By Lemma 7.3.8 we get that $\Gamma \models \cdot = \cdot$, $\Gamma \models \cdot = \cdot : A$, and $\Gamma \models \cdot = \cdot : \Delta$ are partial equivalence relations.

Theorem 7.4.3 (Soundness)**.** $\Gamma \vdash \mathcal{J} \Rightarrow \Gamma \models \mathcal{J}$

The proof of the Soundness Theorem spans the rest of this section. We will mainly state and prove congruence rules as the proof of the other rules are special cases.

Lemma 7.4.4. *The context formation rules are sound:*

$$\frac{}{\diamond \models} \qquad \frac{\Gamma \models \qquad i \notin \mathrm{dom}(\Gamma)}{\Gamma, i : \mathbb{I} \models} \qquad \frac{\Gamma \models \varphi : \mathbb{F}}{\Gamma, \varphi \models} \qquad \frac{\Gamma \models A \qquad x \notin \mathrm{dom}(\Gamma)}{\Gamma, x : A \models}$$

Proof. Immediately by definition. □

Lemma 7.4.5. *Given* $\Gamma \models$, $\Gamma \vdash r : \mathbb{I}$, $\Gamma \vdash s : \mathbb{I}$, $\Gamma \vdash \varphi : \mathbb{F}$, *and* $\Gamma \vdash \psi : \mathbb{F}$ *we have:*

1. $\Gamma \vdash r = s : \mathbb{I} \Rightarrow \Gamma \models r = s : \mathbb{I}$
2. $\Gamma \vdash \varphi = \psi : \mathbb{F} \Rightarrow \Gamma \models \varphi = \psi : \mathbb{F}$

Proof. (1) By virtue of Remark 7.4.2 (3) it is enough to show $r\sigma = s\sigma \in \mathbb{I}(I)$ for $I \Vdash \sigma : \Gamma$. But then by applying the substitution $I \vdash \sigma : \Gamma$ we get $I \vdash r\sigma = s\sigma : \mathbb{I}$, and thus $r\sigma = s\sigma \in \mathbb{I}(I)$ since the context I does not contain restrictions. The proof of (2) is analogous. □

Lemma 7.4.6. *The rule for type conversion is sound:*

$$\frac{\Gamma \models a = b : A \qquad \Gamma \models A = B}{\Gamma \models a = b : B}$$

Proof. Suppose $I \Vdash \sigma = \tau : \Gamma$. By assumption we have $I \Vdash a\sigma = b\tau : A\sigma$. Moreover also $I \Vdash \sigma = \sigma : \Gamma$, so $I \Vdash A\sigma = B\sigma$, and hence $I \Vdash a\sigma = b\tau : B\sigma$ by Lemma 7.3.8 which was what we had to prove. □

Lemma 7.4.7.

$$\frac{\Gamma \models \sigma = \tau : \Delta \qquad \Delta \models A = B}{\Gamma \models A\sigma = B\tau} \qquad \frac{\Gamma \models \sigma = \tau : \Delta \qquad \Delta \models a = b : A}{\Gamma \models a\sigma = b\tau : A\sigma}$$

$$\frac{\Gamma \models \sigma = \tau : \Delta \qquad \Delta \models \delta = \gamma : \Xi}{\Gamma \models \delta\sigma = \gamma\tau : \Xi}$$

Proof. Immediate by definition. □

Lemma 7.4.8. *The rules for* Π*-types are sound:*

1. $$\frac{\Gamma \models A = A' \qquad \Gamma, x : A \models B = B'}{\Gamma \models (x : A) \to B = (x : A') \to B'}$$

2. $$\frac{\Gamma \models A = A' \qquad \Gamma, x : A \models t = t' : B}{\Gamma \models \lambda x : A.t = \lambda x : A'.t' : (x : A) \to B}$$

3. $$\frac{\Gamma \models w = w' : (x : A) \to B \qquad \Gamma \models u = u' : A}{\Gamma \models w\, u = w'\, u' : B(x/u)}$$

4. $$\frac{\Gamma, x : A \models t : B \qquad \Gamma \models u : A}{\Gamma \models (\lambda x : A.t)\, u = t(x/u) : B(x/u)}$$

5. $$\frac{\begin{array}{c}\Gamma \models w : (x : A) \to B \\ \Gamma \models w' : (x : A) \to B \qquad \Gamma, x : A \models w\, x = w'\, x : B\end{array}}{\Gamma \models w = w' : (x : A) \to B}$$

Proof. Abbreviate $(x : A) \to B$ by C. We will make use of Remark 7.3.9.

(1) It is enough to prove this in the case where Γ is of the form I, in which case this directly follows by Pi-E.

(2) Suppose $\Gamma \models A = A'$ and $\Gamma, x : A \models t = t' : B$; this entails $\Gamma, x : A \models B$. For $I \Vdash \sigma = \tau : \Gamma$ we show $I \Vdash (\lambda x : A.t)\sigma = (\lambda x : A'.t')\tau : C\sigma$. For this let $J \Vdash u = v : A\sigma f$ where $f \colon J \to I$. Then also $J \Vdash u = v : A'\tau f$,

$$J \vdash (\lambda x : A.t)\sigma f\, u \succ_{\mathsf{s}} t(\sigma f, x/u) : B(\sigma f, x/u), \text{ and}$$
$$J \vdash (\lambda x : A'.t')\tau f\, v \succ_{\mathsf{s}} t'(\tau f, x/v) : B(\tau f, x/v).$$

Moreover, $J \Vdash (\sigma f, x/u) = (\tau f, x/v) : \Gamma, x : A$, and so $J \Vdash B(\sigma f, x/u) = B(\tau f, x/v)$ and

$$J \Vdash t(\sigma f, x/u) = t'(\tau f, x/v) : B(\sigma f, x/u)$$

which gives

$$J \Vdash (\lambda x : A.t)\sigma f\, u = t(\sigma f, x/u) : B(\sigma f, x/u), \text{ and}$$
$$J \Vdash (\lambda x : A'.t')\tau f\, v = t'(\tau f, x/v) : B(\tau f, x/v),$$

by applying the Expansion Lemma twice, and thus also

$$J \Vdash (\lambda x : A.t)\sigma f\, u = (\lambda x : A'.t')\tau f\, v : B(\sigma f, x/u)$$

what we had to show.

(3) For $I \Vdash \sigma = \tau : \Gamma$ we get $I \Vdash w\sigma = w'\tau : C\sigma$ and $I \Vdash u\sigma = u'\tau : A\sigma$; so also $I \Vdash w\sigma : C\sigma$, therefore $I \Vdash (w\, u)\sigma = w\sigma\, u'\tau = (w'\, u')\tau : B(\sigma, x/u)$.

(4) Given $I \Vdash \sigma = \tau : \Delta$ we get, like in (2), $I \Vdash (\lambda x : A.t)\sigma\, u\sigma = t(\sigma, x/u\sigma) : B(\sigma, x/u\sigma)$ using the Expansion Lemma; moreover, we have $I \Vdash (\sigma, x/u\sigma) = (\tau, x/u\tau) : \Gamma, x : A$, hence

$$I \Vdash (\lambda x : A.t)\sigma\, u\sigma = t(\sigma, x/u\sigma) = t(\tau, x/u\tau) : B(\sigma, x/u\sigma).$$

(5) Suppose $I \Vdash \sigma = \tau : \Gamma$ and $J \Vdash u : A\sigma f$ for $f \colon J \to I$. We have to show $J \Vdash w\sigma f\, u = w'\tau f\, u : B(\sigma f, x/u)$. We have

$$J \Vdash (\sigma f, x/u) = (\tau f, x/u) : \Gamma, x : A$$

and thus, by the assumption $\Gamma, x : A \models w\, x = w'\, x : B$, we get

$$J \Vdash (w\, x)(\sigma f, x/u) = (w'\, x)(\tau f, x/u) : B(\sigma f, x/u).$$

Since x does neither appear in w nor in w' this was what we had to prove. □

Lemma 7.4.9. *The rules for Σ-types are sound:*

1. $$\frac{\Gamma \models A = A' \qquad \Gamma, x : A \models B = B'}{\Gamma \models (x : A) \times B = (x : A') \times B'}$$

2. $$\frac{\Gamma, x : A \models B \qquad \Gamma \models u = u' : A \qquad \Gamma \models v = v' : B(x/u)}{\Gamma \models (u, v) = (u', v') : (x : A) \times B}$$

3. $$\frac{\Gamma, x : A \models B \qquad \Gamma \models w = w' : (x : A) \times B}{\begin{array}{c}\Gamma \models w.1 = w'.1 : A \\ \Gamma \models w.2 = w'.2 : B(x/w.1)\end{array}}$$

4. $$\frac{\Gamma, x : A \models B \qquad \Gamma \models u : A \qquad \Gamma \models v : B(x/u)}{\begin{array}{c}\Gamma \models (u, v).1 = u : A \\ \Gamma \models (u, v).2 = v : B(x/u)\end{array}}$$

5. $$\frac{\begin{array}{c}\Gamma, x : A \models B \qquad \Gamma \models w : (x : A) \times B \qquad \Gamma \models w' : (x : A) \times B \\ \Gamma \models w.1 = w'.1 : A \qquad \Gamma \models w.2 = w'.2 : B(x/w.1)\end{array}}{\Gamma \models w = w' : (x : A) \times B}$$

Lemma 7.4.10. *Given $I, x : \mathsf{N} \models C$ we have:*

1. $$\frac{I \Vdash u : \mathsf{N} \qquad I \Vdash z : C(x/0) \qquad I \Vdash s : (x : \mathsf{N}) \to C \to C(x/\mathsf{S}\,x)}{\begin{array}{c}I \Vdash \mathsf{natrec}\,u\,z\,s : C(x/u) \\ I \Vdash \mathsf{natrec}\,u\,z\,s = (\mathsf{natrec}\,u\,z\,s){\downarrow} : C(x/u)\end{array}}$$

2. $$\frac{\begin{array}{c}I \Vdash u = u' : \mathsf{N} \\ I \Vdash z = z' : C(x/0) \qquad I \Vdash s = s' : (x : \mathsf{N}) \to C \to C(x/\mathsf{S}\,x)\end{array}}{I \Vdash \mathsf{natrec}\,u\,z\,s = \mathsf{natrec}\,u'\,z'\,s' : C(x/u)}$$

Proof. By simultaneous induction on $I \Vdash u : \mathsf{N}$ and $I \Vdash u = u' : \mathsf{N}$.

Case $I \Vdash 0 : \mathsf{N}$. We have $I \vdash \mathsf{natrec}\,0\,z\,s \succ_{\mathsf{s}} z : C(x/0)$ so (1) follows from the Expansion Lemma.

Case $I \Vdash 0 = 0 : \mathsf{N}$. (2) immediately follows from (1) and $I \Vdash z = z' : C(x/0)$.

Case $I \Vdash \mathsf{S}\,u : \mathsf{N}$ from $I \Vdash u : \mathsf{N}$. We have

$$I \vdash \mathsf{natrec}\,(\mathsf{S}\,u)\,z\,s \succ_{\mathsf{s}} s\,u\,(\mathsf{natrec}\,u\,z\,s) : C(x/\mathsf{S}\,u)$$

and $I \Vdash s\,u\,(\mathsf{natrec}\,u\,z\,s) : C(x/\mathsf{S}\,u)$ by IH, and using that u and s are computable. Hence we are done by the Expansion Lemma.

Case $I \Vdash \mathsf{S}\,u = \mathsf{S}\,u' : \mathsf{N}$ from $I \Vdash u = u' : \mathsf{N}$. (2) follows from (1) and $I \Vdash s = s' : (x : \mathsf{N}) \to C \to C(x/\mathsf{S}\,x)$, $I \Vdash u = u' : \mathsf{N}$, and the IH.

Case $I \Vdash u : \mathsf{N}$ for u non-introduced. For $f : J \to I$ we have

$$J \vdash (\mathsf{natrec}\,u\,z\,s)f \succ \mathsf{natrec}\,(uf{\downarrow})\,zf\,sf : C(f, x/uf{\downarrow}).$$

Moreover, we have $I \Vdash uf{\downarrow}$ and $I \Vdash uf{\downarrow} = u{\downarrow}f : \mathsf{N}$ with a shorter derivation (and thus also $J \Vdash C(f, x/uf{\downarrow}) = C(x/u{\downarrow})f$), hence by IH

$$\begin{array}{l}J \Vdash \mathsf{natrec}\,(uf{\downarrow})\,zf\,sf : C(x/u{\downarrow})f, \text{ and} \\ J \Vdash \mathsf{natrec}\,(uf{\downarrow})\,zf\,sf = (\mathsf{natrec}\,(u{\downarrow})\,z\,s)f : C(x/u{\downarrow})f,\end{array}$$

which yields the claim by the Expansion Lemma.

Case $I \Vdash u = u' : \mathsf{N}$ for u or u' non-introduced. We have

$$I \Vdash \mathsf{natrec}\, u\, z\, s = \mathsf{natrec}\, (u{\downarrow})\, z\, s : C(x/u)$$

by either (1) (if u is non-introduced) or by reflexivity (if u is an introduction); likewise for u'. So with the IH for $I \Vdash u{\downarrow} = u'{\downarrow} : \mathsf{N}$ we obtain

$$I \Vdash \mathsf{natrec}\, u\, z\, s = \mathsf{natrec}\, (u{\downarrow})\, z\, s = \mathsf{natrec}\, (u'{\downarrow})\, z'\, s' = \mathsf{natrec}\, u'\, z'\, s' : C(x/u)$$

what we had to show. □

We write $\underline{n}$ for the numeral $\mathsf{S}^n\, 0$ where $n \in \mathbb{N}$.

Lemma 7.4.11. *If* $I \Vdash u : \mathsf{N}$*, then* $I \Vdash u = \underline{n} : \mathsf{N}$ *(and hence also* $I \vdash u = \underline{n} : \mathsf{N}$*) for some* $n \in \mathbb{N}$*.*

Proof. By induction on $I \Vdash u : \mathsf{N}$. The cases for zero and successor are immediate. In case u is non-introduced, then $I \Vdash u{\downarrow} = \underline{n}$ for some $n \in \mathbb{N}$ by IH. By Lemma 7.3.10 (2) and transitivity we conclude $I \Vdash u = \underline{n} : \mathsf{N}$. □

Lemma 7.4.12. $I \Vdash \cdot = \cdot : \mathsf{N}$ *is discrete, i.e., if* $I \Vdash u : \mathsf{N}$*,* $I \Vdash v : \mathsf{N}$*, and* $J \Vdash uf = vg : \mathsf{N}$ *for some* $f, g \colon J \to I$*, then* $I \Vdash u = v : \mathsf{N}$*.*

Proof. By Lemma 7.4.11, we have $I \Vdash u = \underline{n} : \mathsf{N}$ and $I \Vdash v = \underline{m} : \mathsf{N}$ for some $n, m \in \mathbb{N}$, and thus $J \Vdash \underline{n} = uf = vg = \underline{m} : \mathsf{N}$, i.e., $J \Vdash \underline{n} = \underline{m} : \mathsf{N}$ and hence $n = m$ which yields $I \Vdash u = v : \mathsf{N}$. □

Lemma 7.4.13. *The rules for* Path*-types are sound:*

1. $$\frac{\Gamma \models A = A' \qquad \Gamma \models u = u' : A \qquad \Gamma \models v = v' : A}{\Gamma \models \mathsf{Path}\, A\, u\, v = \mathsf{Path}\, A'\, u'\, v'}$$

2. $$\frac{\Gamma \models A \qquad \Gamma, i : \mathbb{I} \models t = t' : A}{\Gamma \models \langle i \rangle t = \langle i \rangle t' : \mathsf{Path}\, A\, t(i0)\, t(i1)}$$

3. $$\frac{\Gamma \models w = w' : \mathsf{Path}\, A\, u\, v \qquad \Gamma \models r = r' : \mathbb{I}}{\Gamma \models w\, r = w'\, r' : A}$$

4. $$\frac{\Gamma \models w : \mathsf{Path}\, A\, u\, v}{\Gamma \models w\, 0 = u : A \qquad \Gamma \models w\, 1 = v : A}$$

5. $$\frac{\Gamma \models A \qquad \Gamma, i : \mathbb{I} \models t : A \qquad \Gamma \models r : \mathbb{I}}{\Gamma \models (\langle i \rangle t)\, r = t(i/r) : A}$$

6. $$\frac{\Gamma \models w : \mathsf{Path}\, A\, u\, v \qquad \Gamma \models w' : \mathsf{Path}\, A\, u\, v \qquad \Gamma, i : \mathbb{I} \models w\, i = w'\, i : A}{\Gamma \models w = w' : \mathsf{Path}\, A\, u\, v}$$

Proof. (1) Follows easily by definition.

(2) For $I \Vdash \sigma = \sigma' : \Gamma$ we have to show

$$I \Vdash (\langle i \rangle t)\sigma = (\langle i \rangle t')\sigma' : \mathsf{Path}\, A\sigma\, t(\sigma, i/0)\, t(\sigma, i/1). \tag{7.10}$$

For $f \colon J \to I$ and $r \in \mathbb{I}(J)$ we have $J \Vdash (\sigma f, i/r) = (\sigma' f, i/r) : \Gamma, i : \mathbb{I}$ and

$$J \vdash (\langle i \rangle t)(\sigma f)\, r \succ_{\mathsf{s}} t(\sigma f, i/r) : A\sigma f, \text{ and}$$
$$J \vdash (\langle i \rangle t')(\sigma' f)\, r \succ_{\mathsf{s}} t'(\sigma' f, i/r) : A\sigma' f,$$

and moreover $J \Vdash t(\sigma f, i/r) = t'(\sigma' f, i/r) : A\sigma f$ and $J \Vdash A\sigma f = A\sigma' f$ by assumption. Hence the Expansion Lemma yields

$$J \Vdash (\langle i \rangle t)(\sigma f)\, r = t(\sigma f, i/r) : A\sigma f, \text{ and}$$
$$J \Vdash (\langle i \rangle t')(\sigma' f)\, r = t'(\sigma' f, i/r) : A\sigma f,$$

in particular also, say $J \Vdash (\langle i \rangle t)\sigma\, 0 = t(\sigma, i/0) : A\sigma$ and $J \Vdash (\langle i \rangle t')\sigma' 0 = t'(\sigma', i/0) = t(\sigma, i/0) : A\sigma$. And hence (7.10) follows.

(3) Supposing $I \Vdash \sigma = \sigma' : \Gamma$ we have to show

$$I \Vdash (w\sigma)\,(r\sigma) = (w'\sigma')\,(r'\sigma) : A\sigma.$$

We have $I \Vdash w\sigma = w'\sigma' : \mathsf{Path}\, A\sigma\, u\sigma\, v\sigma$ and $r\sigma = r'\sigma'$, hence the claim follows by definition.

(4) Let $I \Vdash \sigma = \sigma' : \Gamma$; we have to show, say, $I \Vdash w\sigma\, 0 = u\sigma' : A\sigma$. First, we get $I \Vdash w\sigma : \mathsf{Path}\, A\sigma\, u\sigma\, v\sigma$. Since $\Gamma \models w : \mathsf{Path}\, A\, u\, v$ we also have $\Gamma \models \mathsf{Path}\, A\, u\, v$, hence

$$I \Vdash \mathsf{Path}\, A\sigma\, u\sigma\, v\sigma = \mathsf{Path}\, A\sigma'\, u\sigma'\, v\sigma'. \tag{7.11}$$

Hence we also obtain $I \Vdash w\sigma : \mathsf{Path}\, A\sigma'\, u\sigma'\, v\sigma'$, and thus $I \Vdash w\sigma\, 0 = u\sigma' : A\sigma'$. But (7.11) also yields $I \Vdash A\sigma = A\sigma'$ by definition, so $I \Vdash w\sigma\, 0 = u\sigma' : A\sigma$ what we had to show.

(5) Similar to (2) using the Expansion Lemma.

(6) For $I \Vdash \sigma = \sigma' : \Gamma$, $f \colon J \to I$, and $r \in \mathbb{I}(J)$, we have $J \Vdash (\sigma f, i/r) = (\sigma' f, i/r) : \Gamma, i : \mathbb{I}$, and thus

$$J \Vdash (w\, i)(\sigma f, i/r) = (w'\, i)(\sigma' f, i/r) : A\sigma f. \tag{7.12}$$

But $(w\, i)(\sigma f, i/r)$ is $w\sigma f\ r$, and $(w'\, i)(\sigma' f, i/r)$ is $w'\sigma' f\ r$, so (7.12) is what we had to show. □

Lemma 7.4.14. *Let $\varphi_i \in \mathbb{F}(I)$ and $\varphi_1 \vee \cdots \vee \varphi_n = 1$.*

1. *Let $I, \varphi_i \Vdash_\ell A_i$ and $I, \varphi_i \wedge \varphi_j \Vdash_\ell A_i = A_j$ for all i, j; then*
   - *(a) $I \Vdash_\ell [\varphi_1\ A_1, \ldots, \varphi_n\ A_n]$, and*
   - *(b) $I \Vdash_\ell [\varphi_1\ A_1, \ldots, \varphi_n\ A_n] = A_k$ whenever $\varphi_k = 1$.*

2. *Let* $I \Vdash_\ell A$, $I, \varphi_i \Vdash_\ell t_i : A$, *and* $I, \varphi_i \wedge \varphi_j \Vdash_\ell t_i = t_j : A$ *for all* i, j; *then*

   (a) $I \Vdash_\ell [\varphi_1\ t_1, \ldots, \varphi_n\ t_n] : A$, *and*

   (b) $I \Vdash_\ell [\varphi_1\ t_1, \ldots, \varphi_n\ t_n] = t_k : A$ *whenever* $\varphi_k = 1$.

Proof. (1) Let us abbreviate $[\varphi_1\ A_1, \ldots, \varphi_n\ A_n]$ by A. Because A is non-introduced, we have to show $J \Vdash Af{\downarrow}$ and $J \Vdash Af{\downarrow} = A{\downarrow}f$. For the former observe that $Af{\downarrow}$ is $A_k f$ with k minimal such that $\varphi_k f = 1$. For the latter use that $J \Vdash A_k f = A_l f$ if $\varphi_k f = 1$ and $\varphi_l = 1$, since $I, \varphi_k \wedge \varphi_l \Vdash A_k = A_l$.

(2) Let us write t for $[\varphi_1\ t_1, \ldots, \varphi_n\ t_n]$. By virtue of the Expansion Lemma, it suffices to show $J \Vdash tf{\downarrow} : Af$ and $K \Vdash tf{\downarrow} = t{\downarrow}f : Af$. The proof is just like the proof for types given above. □

Lemma 7.4.15. *Given* $\Gamma \models \varphi_1 \vee \cdots \vee \varphi_n = 1 : \mathbb{F}$, *then:*

$$\frac{\Gamma, \varphi_1 \models \mathcal{J} \ \ldots\ \Gamma, \varphi_n \models \mathcal{J}}{\Gamma \models \mathcal{J}}$$

Proof. Let $\varphi = \varphi_1 \vee \cdots \vee \varphi_n$. Say if $\mathcal{J}$ is a typing judgment of the form A. For $I \Vdash \sigma : \Gamma$ we have $\varphi\sigma = 1$, so $\varphi_k\sigma = 1$ for some k, hence $I \Vdash A\sigma$ by $\Gamma, \varphi_k \models A$. Now let $I \Vdash \sigma = \tau : \Gamma$; then $\varphi_i\sigma = \varphi_i\tau$ (σ and τ assign the same elements to the interval variables), so $\varphi\sigma = \varphi\tau = 1$ yields $\varphi_k\sigma = \varphi_k\tau = 1$ for some common k and thus $I \Vdash A\sigma = A\tau$ follows from $\Gamma, \varphi_k \models A$. The other judgment forms are similar. □

For $I \Vdash A$ and $I, \varphi \Vdash v : A$ we write $I \Vdash u : A[\varphi \mapsto v]$ for $I \Vdash u : A$ and $I, \varphi \Vdash u = v : A$. And likewise $I \Vdash u = w : A[\varphi \mapsto v]$ means $I \Vdash u = w : A$ and $I, \varphi \Vdash u = v : A$ (in this case also $I, \varphi \Vdash w = v : A$ follows). We use similar notations for for "$\models$".

Lemma 7.4.16. *Given* $\varphi \in \mathbb{F}(I)$ *and* $I \Vdash_\ell A, \varphi \in \mathbb{F}(I), I, \varphi \Vdash_\ell T$, *and* $I, \varphi \Vdash_\ell w : \mathsf{Equiv}\, T\, A$, *and write* B *for* $\mathsf{Glue}\, [\varphi \mapsto (T, w)]\, A$. *Then:*

1. $I \Vdash_\ell B$ *and* $I, \varphi \Vdash_\ell B = T$.

2. *If* $I \Vdash_\ell A = A'$, $I, \varphi \Vdash_\ell T = T'$, $I, \varphi \Vdash_\ell w = w' : \mathsf{Equiv}\, T\, A$, *then* $I \Vdash_\ell B = \mathsf{Glue}\, [\varphi \mapsto (T', w')]\, A'$.

3. *If* $I \Vdash_\ell u : B$ *and* $I, \varphi \Vdash_\ell w = w' : \mathsf{Equiv}\, T\, A$, *then* $I \Vdash_\ell \mathsf{unglue}\, [\varphi \mapsto w']\, u : A[\varphi \mapsto w'.1\, u]$ *and* $I \Vdash_\ell \mathsf{unglue}\, [\varphi \mapsto w]\, u = \mathsf{unglue}\, [\varphi \mapsto w']\, u : A$.

4. *If* $I \Vdash_\ell u = u' : B$, *then*

$$I \Vdash_\ell \mathsf{unglue}\, [\varphi \mapsto w]\, u = \mathsf{unglue}\, [\varphi \mapsto w]\, u' : A.$$

5. *If* $I, \varphi \Vdash_\ell t = t' : T$ *and* $I \Vdash_\ell a = a' : A[\varphi \mapsto w.1\, t]$, *then*

   (a) $I \Vdash_\ell \mathsf{glue}\, [\varphi \mapsto t]\, a = \mathsf{glue}\, [\varphi \mapsto t']\, a' : B$,

   (b) $I, \varphi \Vdash_\ell \mathsf{glue}\, [\varphi \mapsto t]\, a = t : T$, *and*

(c) $I \Vdash_\ell \mathsf{unglue}\,[\varphi \mapsto w]\,(\mathsf{glue}\,[\varphi \mapsto t]\,a) = a : A$.

6. If $I \Vdash_\ell u : B$, *then* $I \Vdash_\ell u = \mathsf{glue}\,[\varphi \mapsto u](\mathsf{unglue}\,[\varphi \mapsto w]\,u) : B$.

Proof. (1) Let us first prove $I, \varphi \Vdash B$ and $I, \varphi \Vdash B = T$; but in I, φ, φ becomes 1 so w.l.o.g. let us assume $\varphi = 1$; then B is non-introduced and $I \vdash B \succ_{\mathsf{s}} T$ so $I \Vdash B$ from $I \Vdash T$. For $I \Vdash B = T$ we have to show $J \Vdash Bf{\downarrow} = Tf{\downarrow}$ for $f\colon J \to I$. But $Bf{\downarrow}$ is Tf so this is an instance of Lemma 7.3.10.

It remains to prove $I \Vdash B$ in case where $\varphi \neq 1$; for this use GL-C with the already proven $I, \varphi \Vdash B$.

(2) In case $\varphi \neq 1$ we only have to show $I, \varphi \Vdash B = B'$ and can apply GL-E. But restricted to I, φ, φ becomes 1 and hence we only have to prove the statement for $\varphi = 1$. But then by (1) we have $I \Vdash B = T = T' = B'$.

(3) In case $\varphi \neq 1$, $I \Vdash \mathsf{unglue}\,[\varphi \mapsto w']\,u : A$ and

$$I \Vdash_\ell \mathsf{unglue}\,[\varphi \mapsto w]\,u = \mathsf{unglue}\,[\varphi \mapsto w']\,u : A \tag{7.13}$$

are immediate by definition. Using the Expansion Lemma (and the reduction $I \vdash \mathsf{unglue}\,[\varphi \mapsto w']\,u \succ_{\mathsf{s}} w'.1\,u : A$ for $\varphi = 1$) we obtain $I, \varphi \Vdash \mathsf{unglue}\,[\varphi \mapsto w']\,u = w'.1\,u : A$, which also shows $I \Vdash \mathsf{unglue}\,[\varphi \mapsto w']\,u : A$ as well as (7.13) in case $\varphi = 1$.

(4) In case $\varphi \neq 1$, this is by definition. For $\varphi = 1$ we have

$$I \Vdash \mathsf{unglue}\,[\varphi \mapsto w]\,u = w.1\,u = w.1\,u' = \mathsf{unglue}\,[\varphi \mapsto w]\,u' : A.$$

(5) Let us write b for $\mathsf{glue}\,[\varphi \mapsto t]\,a$, and b' for $\mathsf{glue}\,[\varphi \mapsto t']\,a'$. We first show $I \Vdash b : B$ and $I, \varphi \Vdash b = t : B$ (similarly for b').

In case $\varphi = 1$, $I \vdash b \succ_{\mathsf{s}} t : T$ so by the Expansion Lemma $I \Vdash b : T$ and $I \Vdash b = t : T$, and hence also $I \Vdash b : B$ and $I \Vdash b = t : B$ by (1). This also proves (5b).

Let now φ be arbitrary; we claim

$$I \Vdash \mathsf{unglue}\,[\varphi \mapsto w]\,b : A \text{ and } I \Vdash \mathsf{unglue}\,[\varphi \mapsto w]\,b = a : A$$

(and thus proving (5c)). We will apply the Expansion Lemma to do so; for $f\colon J \to I$ let us analyze the reduct of $(\mathsf{unglue}\,[\varphi \mapsto w]\,b)f$:

$$(\mathsf{unglue}\,[\varphi \mapsto w]\,b)f{\downarrow} = \begin{cases} wf.1\,bf & \text{if } \varphi f = 1, \\ af & \text{otherwise.} \end{cases}$$

Note that, if $\varphi f = 1$, we have as in the case for $\varphi = 1$, $J \Vdash bf = tf : Bf$ and hence $J \Vdash wf.1\,bf = wf.1\,tf = af : Af$. This ensures $J \Vdash (\mathsf{unglue}\,[\varphi \mapsto w]\,b)f{\downarrow} = (\mathsf{unglue}\,[\varphi \mapsto w]\,b){\downarrow}f : A$, and thus the Expansion Lemma applies and we obtain $I \Vdash \mathsf{unglue}\,[\varphi \mapsto w]\,b = (\mathsf{unglue}\,[\varphi \mapsto w]\,b){\downarrow} : A$; but as we have seen in either case, $\varphi = 1$ or not, $I \Vdash (\mathsf{unglue}\,[\varphi \mapsto w]\,b){\downarrow} = a : A$ proving the claim.

Let now be $\varphi \neq 1$, $f\colon J \to I$, and $J \Vdash w' = wf : \mathsf{Equiv}\, Tf\, Af$. We can use the claim for Bf and $\mathsf{Glue}\,[\varphi f \mapsto (Tf, w')]\, Af$ (which is forced equal to Bf by (2)) and obtain both

$$J \Vdash \mathsf{unglue}\,[\varphi f \mapsto wf]\, bf = af : Af \text{ and } J \Vdash \mathsf{unglue}\,[\varphi f \mapsto w']\, bf = af : Af,$$

so the left-hand sides are equal; moreover, $I, \varphi \Vdash b : B$ (as in the case $\varphi = 1$), and hence $I \Vdash b : B$. Likewise one shows $I \Vdash b' : B$.

It remains to show $I \Vdash b = b' : B$. If $\varphi = 1$, we already showed $I \Vdash b = t : T$ and $I \Vdash b' = t' : T$, so the claim follows from $I \Vdash t = t' : T$ and $I \Vdash T = B$. Let us now assume $\varphi \neq 1$. We immediately get $I, \varphi \Vdash b = t = t' = b' : B$ as for $\varphi = 1$. Moreover, we showed above that $I \Vdash \mathsf{unglue}\,[\varphi \mapsto w]\, b = a : A$ and $I \Vdash \mathsf{unglue}\,[\varphi \mapsto w]\, b' = a' : A$. Hence we obtain

$$I \Vdash \mathsf{unglue}\,[\varphi \mapsto w]\, b = \mathsf{unglue}\,[\varphi \mapsto w]\, b' : A$$

from $I \Vdash a = a' : A$.

(6) In case $\varphi = 1$, this follows from (5b). In case $\varphi \neq 1$, we have to show

$$I \Vdash \mathsf{unglue}\,[\varphi \mapsto w]\, u = \mathsf{unglue}\,[\varphi \mapsto w]\, (\mathsf{glue}\,[\varphi \mapsto u](\mathsf{unglue}\,[\varphi \mapsto w]\, u)) : A$$
$$\text{and } I, \varphi \Vdash u = \mathsf{glue}\,[\varphi \mapsto u](\mathsf{unglue}\,[\varphi \mapsto w]\, u) : T.$$

The former is an instance of (5c); the latter follows from (5b). □

Lemma 7.4.17. *Let B be $\mathsf{Glue}\,[\varphi \mapsto (T, w)]\, A$ and suppose $I \Vdash B$ is derived via* Gl-C*, then also $I, \varphi \Vdash T$ and the derivations of $I, \varphi \Vdash T$ are all proper sub-derivations of $I \Vdash B$ (and hence shorter).*

Proof. We have the proper sub-derivations $I, \varphi \Vdash B$. For each $f\colon J \to I$ with $\varphi f = 1$, we have that Bf is non-introduced with reduct Tf so the derivation of $J \Vdash Bf$ has a derivation of $J \Vdash Tf$ as sub-derivation according to Ni-C. □

For the next proof we need a small syntactic observation. Given $\Gamma \vdash \alpha : \mathbb{F}$ irreducible, there is an associated substitution $\bar{\alpha}\colon \Gamma_\alpha \to \Gamma$ where Γ_α skips the names of α and applies a corresponding $\bar{\alpha}$ to the types and restrictions (e.g., if Γ is $i : \mathbb{I}, x : A, j : \mathbb{I}, \varphi$ and α is $(i = 0)$, then Γ_α is $x : A(i0), j : \mathbb{I}, \varphi(i0)$). Since $\alpha\bar{\alpha} = 1$ we even have $\bar{\alpha}\colon \Gamma_\alpha \to \Gamma, \alpha$. The latter has an inverse (w.r.t. judgmental equality) given by the projection $\mathsf{p}\colon \Gamma, \alpha \to \Gamma_\alpha$ (i.e., p assigns each variable in Γ_α to itself): in the context Γ, α, $\bar{\alpha}\mathsf{p}$ is the identity, and $\mathsf{p}\bar{\alpha}$ is the identity since the variables in Γ_α are not changed by $\bar{\alpha}$.

Remark 7.4.18. We can use the above observation to show that the condition $I, \varphi \vdash \mathcal{J}$ in the definition of $I, \varphi \Vdash_\ell \mathcal{J}$ (in Section 7.3) already follows from the other, i.e., $J \Vdash_\ell \mathcal{J}f$ for all $f\colon J \to I, \varphi$: We have to show $I, \alpha \vdash \mathcal{J}$ for each irreducible $\alpha \leq \varphi$. But we have $I_\alpha \Vdash_\ell \mathcal{J}\bar{\alpha}$ by the assumption and $\bar{\alpha}\colon I_\alpha \to I, \varphi$, and hence $I_\alpha \vdash \mathcal{J}\bar{\alpha}$. Substituting along $\mathsf{p}\colon I, \alpha \to I_\alpha$ yields $I, \alpha \vdash \mathcal{J}$.

Theorem 7.4.19. *Compositions are computable, i.e., for $\varphi \in \mathbb{F}(I)$ and $i \notin \mathrm{dom}(I)$:*

1. $$\frac{I, i \Vdash A \qquad I, i, \varphi \Vdash u : A \qquad I \Vdash u_0 : A(i0)[\varphi \mapsto u(i0)]}{I \Vdash \mathsf{comp}^i\, A\, [\varphi \mapsto u]\, u_0 : A(i1)[\varphi \mapsto u(i1)]}$$
$$I \Vdash \mathsf{comp}^i\, A\, [\varphi \mapsto u]\, u_0 = (\mathsf{comp}^i\, A\, [\varphi \mapsto u]\, u_0)\!\downarrow : A(i1)$$

2. $$\frac{I, i \Vdash A \qquad I, i, \varphi \Vdash u = v : A \qquad I \Vdash u_0 = v_0 : A(i0)[\varphi \mapsto u(i0)]}{I \Vdash \mathsf{comp}^i\, A\, [\varphi \mapsto u]\, u_0 = \mathsf{comp}^i\, A\, [\varphi \mapsto v]\, v_0 : A(i1)}$$

3. $$\frac{I, i \Vdash A = B \qquad I, i, \varphi \Vdash u : A \qquad I \Vdash u_0 : A(i0)[\varphi \mapsto u(i0)]}{I \Vdash \mathsf{comp}^i\, A\, [\varphi \mapsto u]\, u_0 = \mathsf{comp}^i\, B\, [\varphi \mapsto u]\, u_0 : A(i1)}$$

Proof. By simultaneous induction on $I, i \Vdash A$ and $I, i \Vdash A = B$. Let us abbreviate $\mathsf{comp}^i\, A\, [\varphi \mapsto u]\, u_0$ by u_1, and $\mathsf{comp}^i\, A\, [\varphi \mapsto v]\, v_0$ by v_1. The second conclusion of (1) holds since in each case we will use the Expansion Lemma and in particular also prove $I \Vdash u_1\!\downarrow : A(i1)$.

Let us first make some preliminary remarks. Given the induction hypothesis holds for $I, i \Vdash A$ we also know that filling operations are admissible for $I, i \Vdash A$, i.e.:

$$\frac{I, i \Vdash A \qquad I, i, \varphi \Vdash u : A \qquad I \Vdash u_0 : A(i0)[\varphi \mapsto u(i0)]}{I, i \Vdash \mathsf{fill}^i\, A\, [\varphi \mapsto u]\, u_0 : A[\varphi \mapsto u, (i = 1) \mapsto u_1]} \tag{7.14}$$

To see this, recall the explicit definition of filling

$$\mathsf{fill}^i\, A\, [\varphi \mapsto u]\, u_0 = \mathsf{comp}^j\, A(i/i \wedge j)\, [\varphi \mapsto u(i/i \wedge j), (i = 0) \mapsto u_0]\, u_0$$

where j is fresh. The derivation of $I, i, j \Vdash A(i/i \wedge j)$ isn't higher than the derivation of $I, i \Vdash A$ so we have to check, with $u' = [\varphi\ u(i/i \wedge j), (i = 0)\ u_0]$ and $A' = A(i/i \wedge j)$,

$$I, i, j, \varphi \vee (i = 0) \Vdash u' : A' \text{ and } I, i, \varphi \vee (i = 0) \Vdash u'(j0) = u_0 : A(i0). \tag{7.15}$$

To check the former, we have to show

$$I, i, j, \varphi \wedge (i = 0) \Vdash u(i/i \wedge j) = u_0 : A'$$

in order to apply Lemma 7.4.14. So let $f \colon J \to I, i, j$ with $\varphi f = 1$ and $f(i) = 0$; then as φ doesn't contain i and j, also $\varphi(f - i, j) = 1$ for $f - i, j \colon J \to I$ being the restriction of f, so by assumption $J \Vdash u(i0)(f - i, j) = u_0(f - i, j) : A(i0)(f - i, j)$. Clearly, $(i0)(f - i, j) = (i/i \wedge j)f$ so the claim follows.

Let us now check the right-hand side equation of (7.15): by virtue of Lemma 7.4.15 we have to check the equation in the contexts I, i, φ and $I, i, (i = 0)$; but $I, i, \varphi \Vdash u'(j0) = u(i0) = u_0 : A(i0)$ and $I, i, (i = 0) \Vdash u'(j0) = u_0 : A(i0)$ by Lemma 7.4.14.

And likewise the filling operation preserves equality.

Case N-C. First, we prove that

$$I, \varphi, i : \mathbb{I} \vdash u = u_0 : \mathsf{N}. \tag{7.16}$$

To show this, it is enough to prove $I, \alpha, i : \mathbb{I} \vdash u = u_0 : \mathsf{N}$ for each $\alpha \leq \varphi$ irreducible. Let $\bar{\alpha} \colon I_\alpha \to I$ be the associated face substitution. We have $I_\alpha, i \Vdash u(\bar{\alpha}, i/i) : \mathsf{N}$ and also $I_\alpha \Vdash u(\bar{\alpha}, i/0) = u_0\bar{\alpha} : \mathsf{N}$ since $\varphi\bar{\alpha} = 1$. By discreteness of N (Lemma 7.4.12),

$$I_\alpha, i \Vdash u(\bar{\alpha}, i/i) = u_0\bar{\alpha} : \mathsf{N},$$

therefore $I_\alpha, i \vdash u(\bar{\alpha}, i/i) = u_0\bar{\alpha} : \mathsf{N}$, i.e., $I_\alpha, i \vdash u\bar{\alpha} = u_0\bar{\alpha} : \mathsf{N}$ with $\bar{\alpha}$ considered as substitution $I_\alpha, i \to I, i$ and u_0 weakened to I, i. Hence $I, \alpha, i : \mathbb{I} \vdash u = u_0 : \mathsf{N}$ by the observation preceding the statement of the theorem.

Second, we prove that

$$I, \varphi \Vdash u(i1) = u_0 : \mathsf{N}. \tag{7.17}$$

$I, \varphi \vdash u(i1) = u_0 : \mathsf{N}$ immediately follows from (7.16). For $f \colon J \to I$ with $\varphi f = 1$ we have to show $J \Vdash u(i1)f = u_0 f : \mathsf{N}$; since $\varphi f = 1$ we get $J \Vdash u(i0)f = u_0 f : \mathsf{N}$ by assumption, i.e., $J \Vdash u(f, i/j)(j0) = u_0(f, i/j)(j0) : \mathsf{N}$ (where u_0 is weakened to I, j and j fresh). By discreteness of N, we obtain $J, j \Vdash u(f, i/j) = u_0(f, i/j) : \mathsf{N}$ and hence $J \Vdash u(f, i/1) = u_0(f, i/1) : \mathsf{N}$, i.e., $J \Vdash u(i1)f = u_0 f : \mathsf{N}$.

We now prove the statements simultaneously by a side induction on $I \Vdash u_0 : \mathsf{N}$ and $I \Vdash u_0 = v_0 : \mathsf{N}$.

Subcase $I \Vdash 0 : \mathsf{N}$. By (7.16) it follows that $I \vdash u_1 \succ_{\mathsf{s}} 0 : \mathsf{N}$, and hence $I \Vdash u_1 : \mathsf{N}$ and $I \Vdash u_1 = 0 : \mathsf{N}$ by the Expansion Lemma. Thus also $I, \varphi \Vdash u_1 = u(i1) : \mathsf{N}$ by (7.17).

Subcase $I \Vdash \mathsf{S}\, u_0' : \mathsf{N}$ from $I \Vdash u_0' : \mathsf{N}$ with $u_0 = \mathsf{S}\, u_0'$. By (7.16) it follows that

$$I \vdash u_1 \succ_{\mathsf{s}} \mathsf{S}(\mathsf{comp}^i\, \mathsf{N}\, [\varphi \mapsto \mathsf{pred}\, u]\, u_0') : \mathsf{N}.$$

From $I, \varphi \Vdash \mathsf{S}\, u_0' = u(i0) : \mathsf{N}$ we get $I, \varphi \Vdash u_0' = \mathsf{pred}(\mathsf{S}\, u_0') = \mathsf{pred}\, u(i0) : \mathsf{N}$ using Lemma 7.4.10 and thus by SIH, $I \Vdash \mathsf{comp}^i\, \mathsf{N}\, [\varphi \mapsto \mathsf{pred}\, u]\, u_0' : \mathsf{N}[\varphi \mapsto (\mathsf{pred}\, u)(i1)]$; hence $I \Vdash u_1 : \mathsf{N}$ and $I \Vdash u_1 = \mathsf{S}(\mathsf{comp}^i\, \mathsf{N}\, [\varphi \mapsto \mathsf{pred}\, u]\, u_0') : \mathsf{N}$ by the Expansion Lemma. Thus also

$$I, \varphi \Vdash u_1 = \mathsf{S}(\mathsf{pred}\, u(i1)) = \mathsf{S}(\mathsf{pred}(\mathsf{S}\, u_0')) = \mathsf{S}\, u_0' = u(i1) : \mathsf{N}$$

using (7.17).

Subcase u_0 is non-introduced. We use the Expansion Lemma: for each $f \colon J \to I$

$$u_1 f{\downarrow} = \mathsf{comp}^j\, \mathsf{N}\, [\varphi f \mapsto u(f, i/j)]\, (u_0 f{\downarrow})$$

the right-hand side is computable by SIH, and this results in a compatible family of reducts by SIH, since we have $K \Vdash u_0 f{\downarrow} g = u_0 f g{\downarrow} : \mathsf{N}$. Thus we get $I \Vdash u_1 : \mathsf{N}$ and $I \Vdash u_1 = u_1{\downarrow} : \mathsf{N}$. By SIH, $I, \varphi \Vdash u_1{\downarrow} = u(i1) : \mathsf{N}$ and thus also $I, \varphi \Vdash u_1 = u(i1) : \mathsf{N}$.

Subcase $I \Vdash 0 = 0 : \mathsf{N}$. Like above we get that $I \Vdash u_1 = 0 = v_1 : \mathsf{N}$.

Subcase $I \Vdash S u_0' = S v_0' : \mathsf{N}$ from $I \Vdash u_0' = v_0' : \mathsf{N}$. Follows from the SIH $I \Vdash \mathsf{comp}^i\, \mathsf{N}\, [\varphi \mapsto \mathsf{pred}\, u]\, u_0' = \mathsf{comp}^i\, \mathsf{N}\, [\varphi \mapsto \mathsf{pred}\, v]\, v_0' : \mathsf{N}$ like above.

Subcase $I \Vdash u_0 = v_0 : \mathsf{N}$ and u_0 or v_0 is non-introduced. We have to show $J \Vdash u_1 f{\downarrow} = v_1 f{\downarrow} : \mathsf{N}$ for $f\colon J \to I$. We have $J \Vdash u_0 f{\downarrow} = v_0 f{\downarrow} : \mathsf{N}$ with a shorter derivation, thus by SIH

$$J \Vdash \mathsf{comp}^j\, \mathsf{N}\, [\varphi f \mapsto u(f, i/j)]\, (u_0 f{\downarrow}) = \mathsf{comp}^j\, \mathsf{N}\, [\varphi f \mapsto v(f, i/j)]\, (v_0 f{\downarrow}) : \mathsf{N}$$

which is what we had to show.

Case Pi-C. Let us write $(x : A) \to B$ for the type under consideration. (1) In view of the Expansion Lemma, the reduction rule for composition at Π-types (which is closed under substitution), and Lemma 7.4.8 (2) and (5), it suffices to show

$$I, x : A(i1) \models \mathsf{comp}^i\, B(x/\bar{x})\, [\varphi \mapsto u\, \bar{x}]\, (u_0\, \bar{x}(i0)) : B(i1), \text{ and} \tag{7.18}$$

$$I, x : A(i1), \varphi \models \mathsf{comp}^i\, B(x/\bar{x})\, [\varphi \mapsto u\, \bar{x}]\, (u_0\, \bar{x}(i0)) = u(i1)\, x : B(i1), \tag{7.19}$$

where $x' = \mathsf{fill}^i\, A(i/1-i)\, []\, x$ and $\bar{x} = x'(i/1-i)$. By IH, we get $I, x : A(i1), i : \mathbb{I} \models \bar{x} : A$ and $I, x : A(i1) \models \bar{x}(i1) = x : A(i1)$, i.e.,

$$I, x : A(i1), i : \mathbb{I} \models \mathsf{fill}^i\, A(i/1-i)\, []\, x : A(i/1-i), \text{ and} \tag{7.20}$$

$$I, x : A(i1) \models (\mathsf{fill}^i\, A(i/1-i)\, []\, x)(i0) = x : A(i1). \tag{7.21}$$

To see (7.20), let $J \Vdash (f, x/a) = (f, x/b) : I, x : A(i1)$, i.e., $f\colon J \to I$ and $J \Vdash a = b : A(i1)f$; for j fresh, we have $J, j \Vdash A(f, i/1-j)$ (note that $(i1)f = (f, i/1-j)(j0)$) and we get

$$J, j \Vdash \mathsf{fill}^j\, A(f, i/1-j)\, []\, a = \mathsf{fill}^j\, A(f, i/1-j)\, []\, b : A(f, i/1-j)$$

by IH, i.e., $J, j \Vdash x'(f, x/a, i/j) = x'(f, x/a, i/j) : A(f, i/1-j)$, and hence for $r \in \mathbb{I}(J)$

$$J \Vdash x'(f, x/a, i/r) = x'(f, x/b, i/r) : (A(i/1-i))(f, x/a, i/r).$$

Thus we get $I, x : A(i1) \models u_0\, \bar{x}(i0) : B(i0)(x/\bar{x}(i0))$, $I, x : A(i1), \varphi, i : \mathbb{I} \models u\, \bar{x} : B(x/\bar{x})$, and

$$I, x : A(i1), \varphi \models u_0\, \bar{x}(i0) = u(i0)\, \bar{x}(i0) = (u\, \bar{x})(i0) : B(i0)(x/\bar{x}(i0)).$$

And hence again by IH, we obtain (7.18) and (7.19).

(2) Let $f\colon J \to I$ and $J \Vdash a : A(f, i/1)$. Then $J, j \Vdash \bar{a} : A(f, i/j)$ as above and we have to show

$$\begin{aligned} J \Vdash \mathsf{comp}^j\, B(f, x/\bar{a}, i/j)\, [\varphi f \mapsto u(f, i/j)\, \bar{a}]\, (u_0 f\, \bar{a}) \\ = \mathsf{comp}^j\, B(f, x/\bar{a}, i/j)\, [\varphi f \mapsto v(f, i/j)\, \bar{a}]\, (v_0 f\, \bar{a}) : \\ B(f, x/\bar{a}(i1), i/1). \end{aligned} \tag{7.22}$$

But this follows directly from the IH for $J, j \Vdash B(f, x/\bar{a}, i/j)$.

Case Si-C. Let us write $(x : A) \times B$ for the type under consideration. (1) We have

$$I, i, \varphi \Vdash u.1 : A \quad \text{and} \quad I \Vdash u_0.1 : A[\varphi \mapsto u.1]$$

so by IH,

$$I, i \Vdash \mathsf{fill}^i\, A\, [\varphi \mapsto u.1]\, (u_0.1) : A[\varphi \mapsto u.1, (i = 0) \mapsto u_0.1].$$

Let us call the above filler w. Thus we get $I, i \Vdash B(x/w)$,

$$I, i, \varphi \Vdash B(x/u.1) = B(x/w) \quad \text{and} \quad I \Vdash B(x/u_0.1) = (B(x/w))(i0)$$

and hence

$$I, i, \varphi \Vdash u.2 : B(x/w) \quad \text{and} \quad I \Vdash u_0.1 : (B(x/w))(i0)[\varphi \mapsto u.2].$$

The IH yields

$$I \Vdash \mathsf{comp}^i\, B(x/w)\, [\varphi \mapsto u.2]\, (u_0.2) : (B(x/w))(i1)[\varphi \mapsto u.2(i1)];$$

let us write w' for the above. By the reduction rules for composition in Σ-types we get $I \vdash u_1 \succ_{\mathsf{s}} (w(i1), w') : (x : A(i1)) \times B(i1)$ and hence the Expansion Lemma yields

$$I \Vdash u_1 = (w(i1), w') : (x : A(i1)) \times B(i1).$$

Which in turn implies the equality

$$I, \varphi \Vdash u_1 = (w(i1), w') = (u.1(i1), u.2(i1)) = u(i1) : (x : A(i1)) \times B(i1).$$

The proof of (2) uses that all notions defining w and w' preserve equality (by IH), and thus $I \Vdash u_1{\downarrow} = v_1{\downarrow} : (x : A(i1)) \times B(i1)$.

Case Pa-C. Let us write $\mathsf{Path}\, A\, a_0\, a_1$ for the type under consideration. We obtain (for j fresh)

$$\begin{aligned} I, j \Vdash \mathsf{comp}^i\, A\, [(j = 0) \mapsto a_0, (j = 1) \mapsto a_1, \varphi \mapsto u\, j]\, (u_0\, j) : \\ A(i1)[(j = 0) \mapsto a_0(i1), (j = 1) \mapsto a_1(i1), \varphi \mapsto u(i1)\, j] \end{aligned} \quad (7.23)$$

by the IH. Using the Expansion Lemma, the reduction rule for composition at Path-types, and Lemma 7.4.13 (2) this yields

$$I \Vdash u_1 : \mathsf{Path}\, A(i1)\, \tilde{u}(j0)\, \tilde{u}(j1)[\varphi \mapsto \langle j \rangle (u(i1)\, j)]$$

where $\tilde{u}$ is the element in (7.23) and u_1 is $\langle j \rangle \tilde{u}$. But $I \Vdash \tilde{u}(jb) = a_b(i1) : A(i1)$, so $I \Vdash u_1 : \mathsf{Path}\, A(i1)\, a_0(i1)\, a_1(i1)$. Moreover,

$$I, \varphi \Vdash u_1 = \langle j \rangle (u(i1)\, j) = u(i1) : \mathsf{Path}\, A(i1)\, a_0(i1)\, a_1(i1)$$

by the correctness of the η-rule for paths (Lemma 7.4.13 (6)).

Case GL-C. To not confuse with our previous notations, we write ψ for the face formula of u, and write B for $\mathsf{Glue}\,[\varphi \mapsto (T, w)]\,A$.

Thus we are given:

$$\frac{1 \neq \varphi \in \mathbb{F}(I, i) \qquad I, i \Vdash A \qquad I, i, \varphi \Vdash w : \mathsf{Equiv}\,T\,A \qquad I, i, \varphi \Vdash B}{I, i \Vdash B}\ \textsc{Gl-C}$$

and also $I, i, \psi \Vdash u : B$ and $I \Vdash u_0 : B(i0)[\psi \mapsto u(i0)]$. Moreover we have $I, i, \varphi \Vdash T$ with shorter derivations by Lemma 7.4.17. We have to show

(i) $I \Vdash u_1 : B(i1)$, and

(ii) $I, \psi \Vdash u_1 = u(i1) : B(i1)$.

We will be using the Expansion Lemma: let $f : J \to I$ and consider the reducts of $u_1 f{\downarrow}$:

$$u_1 f{\downarrow} = \begin{cases} \mathsf{comp}^j\,T f'\,[\psi f \mapsto u f']\,(u_0 f) & \text{if } \varphi f' = 1, \\ \mathsf{glue}\,[\varphi(i1) f \mapsto t_1 f]\,(a_1 f) & \text{otherwise,} \end{cases}$$

with $f' = (f, i/j)$, and t_1 and a_1 as in the corresponding reduction rule, i.e.:

$$\begin{aligned}
a &= \mathsf{unglue}\,[\varphi \mapsto w]\,u && I, i, \psi \\
a_0 &= \mathsf{unglue}\,[\varphi(i0) \mapsto w(i0)]\,u_0 && I \\
\delta &= \forall i.\varphi && I \\
a'_1 &= \mathsf{comp}^i\,A\,[\psi \mapsto a]\,a_0 && I \\
t'_1 &= \mathsf{comp}^i\,T\,[\psi \mapsto u]\,u_0 && I, \delta \\
\omega &= \mathsf{pres}^i\,w\,[\psi \mapsto u]\,u_0 && I, \delta \\
(t_1, \alpha) &= \mathsf{equiv}\,w(i1)\,[\delta \mapsto (t'_1, \omega), \psi \mapsto (u(i1), \langle j \rangle a'_1)]\,a'_1 && I, \varphi(i1) \\
a_1 &= \mathsf{comp}^j\,A(i1)\,[\varphi(i1) \mapsto \alpha\,j, \psi \mapsto a(i1)]\,a'_1 && I
\end{aligned}$$

First, we have to check $J \Vdash u_1 f{\downarrow} : B(i1)f$. In case $\varphi f' = 1$ this immediately follows from the IH. In case $\varphi f' \neq 1$, this follows from the IH and the previous lemmas ensuring that notions involved in the definition of t_1 and a_1 preserve computability.

Second, we have to check $J \Vdash u_1 f{\downarrow} = u_1{\downarrow} f : B(i1)f$. For this, the only interesting case is when $\varphi f' = 1$; then we have to check that:

$$J \Vdash \mathsf{comp}^j\,T f'\,[\psi f \mapsto u f']\,(u_0 f) = \mathsf{glue}\,[\varphi(i1) f \mapsto t_1 f]\,(a_1 f) : B(i1)f \quad (7.24)$$

Since all the involved notions commute with substitutions, we may (temporarily) assume $f = \mathrm{id}$ and $\varphi = 1$ to simplify notation. Then also $\delta = 1 = \varphi(i1)$, and hence (using the IH)

$$I \Vdash t_1 = t'_1 = \mathsf{comp}^i\,T\,[\psi \mapsto u]\,u_0 : T(i1),$$

so (7.24) follows from Lemma 7.4.16 (5b) and (1).

So the Expansion Lemma yields (i) and $I \Vdash u_1 = \mathsf{glue}\,[\varphi(i1) \mapsto t_1]\,a_1 : B(i1)$. (ii) is checked similarly to what is done in [26, Appendix A] using the IH. This proves (1) in this case; for (2) one uses that all notions for giving a_1 and t_1 above preserve equality, and thus $I \Vdash u_1{\downarrow} = v_1{\downarrow} : B(i1)$ entailing $I \Vdash u_1 = v_1 : B(i1)$.

Case U-C. We have

$$I \vdash \mathsf{comp}^i\,\mathsf{U}\,[\varphi \mapsto u]\,u_0 \succ_{\mathsf{s}} \mathsf{Glue}\,[\varphi \mapsto (u(i1), \mathsf{equiv}^i\,u(i/1-i))]\,u : \mathsf{U}$$

thus it is sufficient to prove that the right-hand side is computable, i.e.,

$$I \Vdash_1 \mathsf{Glue}\,[\varphi \mapsto (u(i1), \mathsf{equiv}^i\,u(i/1-i))]\,u_0 : \mathsf{U}$$

that is,

$$I \Vdash_0 \mathsf{Glue}\,[\varphi \mapsto (u(i1), \mathsf{equiv}^i\,u(i/1-i))]\,u_0.$$

We have $I \Vdash_0 u_0$ so by Lemma 7.4.16 (1) it suffices to prove

$$I \Vdash_0 \mathsf{equiv}^i\,u(i/1-i) : \mathsf{Equiv}\,u(i1)\,u_0.$$

To see this recall that the definition of $\mathsf{equiv}^i\,u(i/1-i)$ is defined from compositions and filling operations for types $I, i \Vdash_0 u$ and $I, i \Vdash_0 u(i/1-i)$ using operations we already have shown to preserve computability. But in this case we have as IH, that these composition and filling operations are computable since the derivations of $I, i, \varphi \Vdash_0 u$ and $I, i, \varphi \Vdash_0 u(i/1-i)$ are less complex than the derivation $I \Vdash_1 \mathsf{U}$ since the level is smaller.

Case Ni-C. So we have $J \Vdash Af{\downarrow}$ for each $f : J \to I, i$ and $J \Vdash A{\downarrow}f = Af{\downarrow}$ (all with a shorter derivation than $I, i \Vdash A$). Note that by Lemma 7.3.10 (1), we also have $I, i \Vdash A = A{\downarrow}$.

(1) We have to show $J \Vdash u_1 f : A(i1)f{\downarrow}$ for each $f : J \to I$. It is enough to show this for f being the identity; we do this using the Expansion Lemma. Let $f : J \to I$ and j be fresh, $f' = (f, i/j)$; we first show $J \Vdash u_1 f{\downarrow} : A{\downarrow}(i1)f$. We have

$$J \vdash u_1 f \succ \mathsf{comp}^j\,(Af'{\downarrow})\,[\varphi f \mapsto uf']\,u_0 f : Af'(j1)$$

hence also at type $Af'(j1){\downarrow}$, and so, by IH (1) for $J, j \Vdash Af'{\downarrow}$, we obtain $J \Vdash u_1 f{\downarrow} : Af'(j1){\downarrow}$. But $J \Vdash Af'(j1){\downarrow} = A{\downarrow}(i1)f$, so $J \Vdash u_1 f{\downarrow} : A{\downarrow}(i1)f$.

Next, we have to show $J \Vdash u_1{\downarrow}f = u_1 f{\downarrow} : A{\downarrow}(i1)f$. Since $J, j \Vdash A{\downarrow}f' = Af'{\downarrow}$ (with a shorter derivation) we get by IH (3), $J \Vdash u_1{\downarrow}f = u_1 f{\downarrow} : A{\downarrow}f'(j1)$ what we had to show.

Thus we can apply the Expansion Lemma and obtain $I \Vdash u_1 : A{\downarrow}(i1)$ and $I \Vdash u_1 = u_1{\downarrow} : A{\downarrow}(i1)$, and hence also $I \Vdash u_1 : A(i1)$ and $I \Vdash u_1 = u_1{\downarrow} : A(i1)$. By IH, we also have $I, \varphi \Vdash u_1 = u_1{\downarrow} = u(i1) : A{\downarrow}(i1) = A(i1)$.

(2) Like above, we obtain

$$I \Vdash u_1 = u_1{\downarrow} : A{\downarrow}(i1) \quad \text{and} \quad I \Vdash v_1 = v_1{\downarrow} : A{\downarrow}(i1).$$

But since the derivation of $I, i \Vdash A{\downarrow}$ is shorter, and $u_1{\downarrow} = \mathsf{comp}^i\,A{\downarrow}\,[\varphi \mapsto u]\,u_0$ and similarly for $v_1{\downarrow}$, the IH yields $I \Vdash u_1{\downarrow} = v_1{\downarrow} : A{\downarrow}(i1)$, thus also $I \Vdash u_1 = v_1 : A{\downarrow}(i1)$, that is, $I \Vdash u_1 = v_1 : A(i1)$ since $I, i \Vdash A = A{\downarrow}$.

It remains to show that composition preserves forced type equality (i.e., (3) holds). The argument for the different cases is very similar, namely using that the compositions on the left-hand and right-hand side of (3) are equal to their respective reducts (by (1)) and then applying the IH for the reducts. We will only present the case Ni-E.

Case Ni-E. Then A or B is non-introduced and $I, i \Vdash A{\downarrow} = B{\downarrow}$ with a shorter derivation. Moreover, by (1) (if the type is non-introduced) or reflexivity (if the type is introduced) we have

$$I \Vdash \mathsf{comp}^i\, A\, [\varphi \mapsto u]\, u_0 = \mathsf{comp}^i\, (A{\downarrow})\, [\varphi \mapsto u]\, u_0 : A(i1), \text{ and}$$
$$I \Vdash \mathsf{comp}^i\, B\, [\varphi \mapsto u]\, u_0 = \mathsf{comp}^i\, (B{\downarrow})\, [\varphi \mapsto u]\, u_0 : B(i1),$$

but the right-hand sides are forced equal by IH. □

Lemma 7.4.20. *The rules for the universe* U *are sound:*

1. $\Gamma \models A : \mathsf{U} \Rightarrow \Gamma \models A$
2. $\Gamma \models A = B : \mathsf{U} \Rightarrow \Gamma \models A = B$

Moreover, the rules reflecting the type formers in U *are sound.*

Proof. Of the first two statements let us only prove (2): given $I \Vdash \sigma = \tau : \Gamma$ we get $I \Vdash A\sigma = B\tau : \mathsf{U}$; this must be a derivation of $I \Vdash_1 A\sigma = B\tau : \mathsf{U}$ and hence we also have $I \Vdash_0 A\sigma = B\tau$.

The soundness of the rules reflecting the type formers in U is proved very similar to proving the soundness of the type formers. Let us exemplify this by showing soundness for Π-types in U: we are give $\Gamma \models A : \mathsf{U}$ and $\Gamma, x : A \models B : \mathsf{U}$, and want to show $\Gamma \models (x : A) \to B : \mathsf{U}$. Let $I \Vdash \sigma = \tau : \Gamma$, then $I \Vdash A\sigma = A\tau : \mathsf{U}$, so, as above, $I \Vdash_0 A\sigma = A\tau$; it is enough to show

$$J \Vdash_0 B(\sigma f, x/u) = B(\tau f, x/v) \tag{7.25}$$

for $J \Vdash u = v : A\sigma f$ with $f : J \to I$. Then $J \Vdash (\sigma f, x/u) = (\tau f, x/v) : \Gamma, x : A$, hence $J \Vdash B(\sigma f, x/u) = B(\tau f, x/v) : \mathsf{U}$ and hence (7.25). □

Proof of Soundness (Theorem 7.4.3). By induction on the derivation $\Gamma \vdash \mathcal{J}$.

We have already seen above that most of the rules are sound. Let us now look at the missing rules. Concerning basic type theory, the formation and introduction rules for N are immediate; its elimination rule and definitional equality follow from the "local" soundness from Lemma 7.4.10 as follows. Suppose $\Gamma \models u : \mathsf{N}$, $\Gamma, x : \mathsf{N} \models C$, $\Gamma \models z : C(x/0)$, and $\Gamma \models s : (x : \mathsf{N}) \to C \to C(x/\mathsf{S}\, x)$. For $I \Vdash \sigma = \tau : \mathsf{N}$ we get by Lemma 7.4.10 (2)

$$I \Vdash \mathsf{natrec}\, u\sigma\, z\sigma\, s\sigma = \mathsf{natrec}\, u\tau\, z\tau\, s\tau : C(\sigma, x/u\sigma).$$

(Hence $\Gamma \models \mathsf{natrec}\, u\, z\, s : C(x/u)$.) Concerning, the definitional equality, if, say, u was of the form $\mathsf{S}\, v$, then, Lemma 7.4.10 (1) gives

$$\begin{aligned} I \Vdash \mathsf{natrec}\,(\mathsf{S}\, v\sigma)\, z\sigma\, s\sigma &= \mathsf{natrec}\,(\mathsf{S}\, v\tau)\, z\tau\, s\tau \\ &= (\mathsf{natrec}\,(\mathsf{S}\, v\tau)\, z\tau\, s\tau)\downarrow : C(\sigma, x/u\sigma). \end{aligned}$$

and $(\mathsf{natrec}\,(\mathsf{S}\, v\tau)\, z\tau\, s\tau)\downarrow$ is $s\tau\, v\tau\, (\mathsf{natrec}\, v\tau\, z\tau\, s\tau)$, proving

$$\Gamma \models \mathsf{natrec}\,(\mathsf{S}\, v)\, z\, s = s\, v\, (\mathsf{natrec}\, v\, z\, s) : C(x/\,\mathsf{S}\, v);$$

similarly, the soundness of the other definitional equality is established.

Let us now look at the composition operations: suppose $\Gamma, i : \mathbb{I} \models A$, $\Gamma \models \varphi : \mathbb{F}$, $\Gamma, \varphi, i : \mathbb{I} \models u : A$, and $\Gamma \models u_0 : A(i0)[\varphi \mapsto u(i0)]$. Further let $I \Vdash \sigma = \tau : \Gamma$, then for j fresh, $I, j \Vdash \sigma' = \tau' : \Gamma, i : \mathbb{I}$ where $\sigma' = (\sigma, i/j)$ and $\tau' = (\tau, i/j)$, hence $I, j \Vdash A\sigma' = A\tau'$, $\varphi\sigma = \varphi\tau$, $I, j, \varphi\sigma \Vdash u\sigma' = u\tau' : A\sigma'$, and $I \Vdash u_0\sigma = u_0\tau : A\sigma'(j0)[\varphi\sigma \mapsto u\sigma'(j0)]$. By Theorem 7.4.19,

$$I \Vdash \mathsf{comp}^j\, (A\sigma')\, [\varphi\sigma \mapsto u\sigma']\, (u_0\sigma) = \mathsf{comp}^j\, (A\tau')\, [\varphi\tau \mapsto u\tau']\, (u_0\tau) : A\sigma'(j1)$$

and

$$I, \varphi\sigma \Vdash \mathsf{comp}^j\, (A\sigma')\, [\varphi\sigma \mapsto u\sigma']\, (u_0\sigma) = u\sigma'(j1) = u\tau'(j1) : A\sigma'(j1)$$

hence we showed $\Gamma \models \mathsf{comp}^i\, A\, [\varphi \mapsto u]\, u_0 : A(i1)[\varphi \mapsto u(i1)]$. Similarly one can justify the congruence rule for composition.

The definitional equalities which hold for comp follow from the second conclusion of Theorem 7.4.19 (1), i.e., that a composition is forced equal to its reduct.

The remaining rules for systems follow from their "local" analogues in form of Lemma 7.4.14; let us, say, suppose $\Gamma \models \varphi_1 \vee \cdots \vee \varphi_n = 1 : \mathbb{F}$, $\Gamma, \varphi_i \models A_i$, and $\Gamma, \varphi_i \wedge \varphi_j \models A_i = A_j$. For $I \Vdash \sigma = \tau : \Gamma$ we get k with $\varphi_k\sigma = \varphi_k\tau = 1$ like in the proof of Lemma 7.4.15 so, writing A for $[\varphi_1\ A_1, \ldots, \varphi_n\ A_n]$,

$$I \Vdash A\sigma = A_k\sigma = A_k\tau = A\tau$$

by Lemma 7.4.14 and using $\Gamma, \varphi_k \models A_k$, so $\Gamma \models A$. Likewise, if $\Gamma \models \varphi_l = 1 : \mathbb{F}$ for some l, then $I \Vdash A\sigma = A_l\sigma = A_l\tau$, showing $\Gamma \models A = A_l$ in this case. The other rules concerning systems are justified similarly.

The soundness of the remaining rules concerning Glue follow similarly from their "local" version in Lemma 7.4.16. □

Corollary 7.4.21 (Canonicity)**.** *If I is a context of the form $i_1 : \mathbb{I}, \ldots, i_k : \mathbb{I}$ and $I \vdash u : \mathsf{N}$, then $I \vdash u = \underline{n} : \mathsf{N}$ for a unique $n \in \mathbb{N}$.*

Proof. By Soundness, $I \models u : \mathsf{N}$ hence $I \Vdash u : \mathsf{N}$, so $I \Vdash u = \underline{n} : \mathsf{N}$ for some $n \in \mathbb{N}$ by Lemma 7.4.11, and thus also $I \vdash u = \underline{n} : \mathsf{N}$. The uniqueness follows since $I \vdash \underline{n} = \underline{m} : \mathsf{N}$ yields $I \Vdash \underline{n} = \underline{m} : \mathsf{N}$ which is only the case for $n = m$. □

Corollary 7.4.22 (Consistency)**.** *Cubical type theory is consistent, i.e., there is a type in the empty context which is not inhabited.*

Proof. Consider the type $\mathsf{Path}\,\mathsf{N}\,0\,1$ and suppose there is a u with $\vdash u : \mathsf{Path}\,\mathsf{N}\,0\,1$. Hence we get $i : \mathbb{I} \vdash u\,i : \mathsf{N}$, as well as $\vdash u\,0 = 0 : \mathsf{N}$ and $\vdash u\,1 = 1 : \mathsf{N}$. By Canonicity, we get $n \in \mathbb{N}$ with $i : \mathbb{I} \vdash u\,i = \underline{n} : \mathsf{N}$, and hence (by substitution) $\vdash u\,0 = \underline{n} : \mathsf{N}$ and $\vdash u\,1 = \underline{n} : \mathsf{N}$, so $\vdash 0 = 1 : \mathsf{N}$, contradicting the uniqueness in Corollary 7.4.21. □

Remark 7.4.23. One could also extend cubical type theory with an empty type N_0 whose forcing relation is empty; consistency for this extension is then an immediate consequence of the corresponding Soundness Theorem.

7.5 Extending with the Circle

In this section we sketch how the proof of canonicity can be extended to the system where a circle S^1 is added; the extension with other spheres is done analogously.

First, we generalize Path-types to dependent path types $\mathsf{Path}^i\,A\,u\,v$ (i might now appear in A); this extension is straightforward, e.g., the β-reduction rule for paths now reads

$$\frac{\Gamma, i : \mathbb{I} \vdash A \qquad \Gamma, i : \mathbb{I} \vdash t : A \qquad \Gamma \vdash r : \mathbb{I}}{\Gamma \vdash (\langle i \rangle t)\,r \succ t(i/r) : A(i/r)}$$

and likewise the computability predicates and relations are easily adapted.

Next, we have to extend the reduction relation as follows to incorporate the circle.

$$\frac{\Gamma \vdash}{\Gamma \vdash \mathsf{loop}\,0 \succ \mathsf{base} : \mathsf{S}^1} \qquad \frac{\Gamma, i : \mathbb{I} \vdash u : \mathsf{S}^1}{\Gamma \vdash \mathsf{comp}^i\,\mathsf{S}^1\,[1 \mapsto u]\,u(i0) \succ u(i1) : \mathsf{S}^1}$$
$$\Gamma \vdash \mathsf{loop}\,1 \succ \mathsf{base} : \mathsf{S}^1$$

(For simplicity, we will use $\mathsf{comp}^i\,\mathsf{S}^1$ instead of adding yet another constructor hcomp^i as was done in in [26].)

Given $\Gamma, x : \mathsf{S}^1 \vdash C$, $\Gamma \vdash b : C(x/\mathsf{base})$ and $\Gamma \vdash l : \mathsf{Path}^i\,C(x/\,\mathsf{loop}\,i)\,b\,b$ we also add the reduction rules for the elimination

$$\Gamma \vdash \mathsf{S}^1\text{-}\mathsf{elim}^x\,\mathsf{base}\,C\,b\,l \succ b : C(x/\mathsf{base})$$
$$\Gamma \vdash \mathsf{S}^1\text{-}\mathsf{elim}^x\,(\mathsf{loop}\,r)\,C\,b\,l \succ l\,r : C(x/\,\mathsf{loop}\,r)$$

where $\Gamma \vdash r \neq 1 : \mathbb{I}$, and moreover for $\Gamma \vdash \varphi \neq 1 : \mathbb{F}$,

$$\Gamma \vdash \mathsf{S}^1\text{-}\mathsf{elim}^x\,(\mathsf{comp}^i\,\mathsf{S}^1\,[\varphi \mapsto u]\,u_0)\,C\,b\,l \succ$$
$$\mathsf{comp}^i\,C(x/v)\,[\varphi \mapsto u']\,u'_0 : C(x/\mathsf{comp}^i\,\mathsf{S}^1\,[\varphi \mapsto u]\,u_0)$$

where $v = \mathsf{fill}^i\,\mathsf{S}^1\,[\varphi \mapsto u]\,u_0$, $u' = \mathsf{S}^1\text{-}\mathsf{elim}^x\,u\,C\,b\,l$, $u'_0 = \mathsf{S}^1\text{-}\mathsf{elim}^x\,u_0\,C\,b\,l$, and we assumed $i \notin \mathrm{dom}\,\Gamma$ (otherwise rename i).

Furthermore, if $\Gamma \vdash t \succ t' : \mathsf{S}^1$, then

$$\Gamma \vdash \mathsf{S}^1\text{-}\mathsf{elim}^x\, t\, C\, b\, l \succ \mathsf{S}^1\text{-}\mathsf{elim}^x\, t'\, C\, b\, l : C(x/t').$$

Consequently, we also call expressions introduced if they are of the form base, $\mathsf{loop}\, r$ with $r \notin \{0,1\}$, and $\mathsf{comp}^i\, \mathsf{S}^1\, [\varphi \mapsto u]\, u_0$ with $\varphi \neq 1$.

Next, the computability predicates and relations are adapted as follows: $I \Vdash_\ell \mathsf{S}^1$ and $I \Vdash_\ell \mathsf{S}^1 = \mathsf{S}^1$. $I \Vdash_\ell u : \mathsf{S}^1$ and $I \Vdash u = v : \mathsf{S}^1$ are defined simultaneously (similarly as for N):

$$\frac{}{I \Vdash_\ell \mathsf{base} : \mathsf{S}^1} \qquad \frac{r \in \mathbb{I}(I) - \{0,1\} \qquad I \Vdash_\ell \mathsf{loop}\, 0 : \mathsf{S}^1 \qquad I \Vdash_\ell \mathsf{loop}\, 1 : \mathsf{S}^1}{I \Vdash_\ell \mathsf{loop}\, r : \mathsf{S}^1}$$

$$\frac{\begin{array}{c} 1 \neq \varphi \in \mathbb{F}(I) \\ I, i, \varphi \Vdash_\ell u : \mathsf{S}^1 \qquad I \Vdash_\ell u_0 = u(i0) : \mathsf{S}^1 \qquad I, \varphi \Vdash_\ell \mathsf{comp}^i\, \mathsf{S}^1\, [\varphi \mapsto u]\, u_0 : \mathsf{S}^1 \end{array}}{I \Vdash_\ell \mathsf{comp}^i\, \mathsf{S}^1\, [\varphi \mapsto u]\, u_0 : \mathsf{S}^1}$$

$$\frac{\begin{array}{c} u \text{ n.i.} \qquad \forall f \colon J \to I (uf!^{\mathsf{S}^1} \;\&\; J \Vdash_\ell uf\downarrow^{\mathsf{S}^1} : \mathsf{S}^1) \\ \forall f \colon J \to I \forall g \colon K \to J (K \Vdash_\ell uf\downarrow g = ufg\downarrow : \mathsf{S}^1) \end{array}}{I \Vdash_\ell u : \mathsf{S}^1}$$

Note, the (admissible) two last premises in the case for loop are there to not increase the height of the derivation when doing a substitution (Lemma 7.3.6); similarly for the last premise in the rule for composition. The relation $I \Vdash_\ell u = v : \mathsf{S}^1$ is defined analogously, that is, by the usual congruence rules and a clause for when u or v is non-introduced as we have it for N.

7.6 Conclusion

We have shown canonicity for cubical type theory [26] and its extension with the circle. This establishes that the judgmental equalities of the theory are sufficient to compute closed naturals to numerals; indeed, we have even given a deterministic reduction relation to do so. It should be noted that we could have also worked with the corresponding *untyped* reduction relation $A \succ B$ and then take $I \vdash A \succ B$ to mean $I \vdash A = B$ and $A \succ B$ etc.

To prove canonicity we devised computability predicates (and relations) which, from a set-theoretic perspective, are constructed using the *least* fixpoint of a suitable operator. It is unlikely that this result is optimal in terms of proof-theoretic strength; we conjecture that it is possible to modify the argument to only require the existence of a fixpoint of a suitably modified operator (and not necessarily its least fixpoint); this should be related to how canonicity is established in [7].

We expect that the present work can be extended to get a normalization theorem and to establish decidability of type checking for cubical type theory (and proving its implementation[2] correct). One new aspect of such an adaption

[2] Available at `https://github.com/mortberg/cubicaltt`.

is to generalize the computability predicates and relations to expressions in any contexts in which we get new introduced expressions given by systems; moreover, we will have to consider reductions in such general contexts as well which has to ensure that, say, variables of path-types compute to the right endpoints.

Another direction of future research is to investigate canonicity of various extensions of cubical type theory, especially adding resizing rules.

Acknowledgments. I thank Carlo Angiuli, Thierry Coquand, Robert Harper, and Bassel Mannaa for discussions about this work.

Bibliography

[1] Andreas Abel, Thierry Coquand, and Bassel Mannaa, *On the decidability of conversion in type theory*, Abstract for TYPES 2016, 2016.

[2] Andreas Abel and Gabriel Scherer, *On irrelevance and algorithmic equality in predicative type theory*, Logical Methods in Computer Science **8** (2012), no. 1, 1–36, TYPES'10 special issue.

[3] Peter Aczel, *On relating type theories and set theories*, Types for Proofs and Programs (T. Altenkirch, B. Reus, and W. Naraschewski, eds.), Lecture Notes in Computer Science, vol. 1657, Springer Verlag, Berlin, Heidelberg, New York, 1999, pp. 1–18.

[4] Thorsten Altenkirch, *Extensional equality in intensional type theory*, 14th Symposium on Logic in Computer Science, 1999, pp. 412–420.

[5] Thorsten Altenkirch and Ambrus Kaposi, *Towards cubical type theory*, Preprint, 2014.

[6] Thorsten Altenkirch, Conor McBride, and Wouter Swierstra, *Observational equality, now!*, PLPV '07: Proceedings of the 2007 workshop on Programming languages meets program verification (New York, NY, USA), ACM, 2007, pp. 57–68.

[7] Carlo Angiuli and Robert Harper, *Computational higher type theory II: Dependent cubical realizability*, Preprint arXiv:1606.09638v1 [cs.LO], 2016.

[8] Carlo Angiuli, Robert Harper, and Todd Wilson, *Computational higher type theory I: Abstract cubical realizability*, Preprint arXiv:1604.08873v1 [cs.LO], 2016.

[9] Steve Awodey and Michael A. Warren, *Homotopy theoretic models of identity types*, Math. Proc. Cambridge Philos. Soc. **146** (2009), no. 1, 45–55.

[10] Raymond Balbes and Philip Dwinger, *Distributive lattices*, University of Missouri Press, 1975.

[11] Bruno Barras, Thierry Coquand, and Simon Huber, *A generalization of the Takeuti-Gandy interpretation*, Mathematical Structures in Computer Science **25** (2015), 1071–1099.

[12] Jean-Philippe Bernardy, Thierry Coquand, and Guilhem Moulin, *A presheaf model of parametric type theory*, Electronic Notes in Theoretical Computer Science **319** (2015), 67–82.

[13] Jean-Philippe Bernardy and Guilhem Moulin, *Type-theory in color*, ACM SIGPLAN International Conference on Functional Programming, ICFP'13, Boston, MA, USA - September 25–27, 2013, pp. 61–72.

[14] Marc Bezem and Thierry Coquand, *A Kripke model for simplicial sets*, Theoretical Computer Science **574** (2015), 86–91.

[15] Marc Bezem, Thierry Coquand, and Simon Huber, *A model of type theory in cubical sets*, 19th International Conference on Types for Proofs and Programs (TYPES 2013) (Dagstuhl, Germany) (Ralph Matthes and Aleksy Schubert, eds.), Leibniz International Proceedings in Informatics (LIPIcs), vol. 26, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2014, pp. 107–128.

[16] Marc Bezem, Thierry Coquand, and Erik Parmann, *Non-constructivity in Kan simplicial sets*, 13th International Conference on Typed Lambda Calculi and Applications (TLCA 2015) (Dagstuhl, Germany) (Thorsten Altenkirch, ed.), Leibniz International Proceedings in Informatics (LIPIcs), vol. 38, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2015, pp. 92–106.

[17] Lars Birkedal, Aleš Bizjak, Ranald Clouston, Hans Bugge Grathwohl, Bas Spitters, and Andrea Vezzosi, *Guarded cubical type theory: Path equality for guarded recursion*, 25th EACSL Annual Conference on Computer Science Logic (CSL 2016) (Dagstuhl, Germany) (Jean-Marc Talbot and Laurent Regnier, eds.), Leibniz International Proceedings in Informatics (LIPIcs), vol. 62, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2016, pp. 23:1–23:17.

[18] Erret Bishop and Douglas S. Bridges, *Constructive analysis*, Grundlehren der mathematischen Wissenschaften, vol. 279, Springer Verlag, Berlin, Heidelberg, New York, 1985.

[19] Ronald Brown, Philip J. Higgins, and Rafael Sivera, *Nonabelian algebraic topology: Filtered spaces, crossed complexes, cubical homotopy groupoids*, EMS Tracts in Mathematics, vol. 15, European Mathematical Society (EMS), Zürich, 2011.

[20] Guillaume Brunerie, *On the homotopy groups of spheres in homotopy type theory*, Ph.D. thesis, Université de Nice, 2016.

[21] Guillaume Brunerie and Daniel R. Licata, *A cubical infinite-dimensional type theory*, Slides of a talk at Oxford Homotopy Type Theory Workshop, November 2014.

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

[23] Alonzo Church, *A formulation of the simple theory of types*, The Journal of Symbolic Logic **5** (1949), no. 2, 56–68.

[24] Denis-Charles Cisinski, *Univalent universes for elegant models of homotopy types*, Preprint arXiv:1406.0058 [math.AT], May 2014.

[25] Cyril Cohen, Thierry Coquand, Simon Huber, and Anders Mörtberg, *Cubical*, `https://github.com/simhu/cubical`, 2013.

[26] ———, *Cubical type theory: a constructive interpretation of the univalence axiom*, to appear in the proceedings of TYPES 2015, 2015.

[27] ———, *Cubicaltt*, `https://github.com/mortberg/cubicaltt`, 2015.

[28] Thierry Coquand, *A property of contractible types*, Unpublished note available at `http://www.cse.chalmers.se/~coquand/contr.pdf`, December 2013.

[29] Thierry Coquand and Nils Anders Danielsson, *Isomorphism is equality*, Indagationes Mathematicae **24** (2013), no. 4, 1105–1120.

[30] Thierry Coquand and Gérard Huet, *The calculus of constructions*, Information and Computation **76** (1988), 95–129.

[31] Thierry Coquand and Bassel Mannaa, *The independence of Markov's principle in type theory*, 1st International Conference on Formal Structures for Computation and Deduction (FSCD 2016) (Dagstuhl, Germany) (Delia Kesner and Brigitte Pientka, eds.), Leibniz International Proceedings in Informatics (LIPIcs), vol. 52, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2016, pp. 17:1–17:18.

[32] Nicolaas G. de Bruijn, *The mathematical language AUTOMATH, its usage, and some of its extensions*, Symposium on Automatic Demonstration (Versailles, 1968), Lecture Notes in Mathematics, vol. 125, Springer Verlag, Berlin, Heidelberg, New York, 1970, pp. 29–61.

[33] Simon Docherty, *A model of type theory in cubical sets with connections*, Master's thesis, Universiteit van Amsterdam, 2014.

[34] Peter Dybjer, *Internal type theory*, Lecture Notes in Computer Science, Springer Verlag, Berlin, Heidelberg, New York, 1996, pp. 120–134.

[35] ———, *A general formulation of simultaneous inductive-recursive definitions in type theory*, The Journal of Symbolic Logic **65** (2000), no. 2, 525–549.

[36] Michael P. Fourman and Dana S. Scott, *Sheaves and logic*, Applications of Sheaves (M. Fourman and D. Scott, eds.), Lecture Notes in Mathematics, vol. 753, Springer Verlag, Berlin, Heidelberg, New York, 1979, pp. 302–401.

[37] Nicola Gambino and Christian Sattler, *Uniform fibrations and the Frobenius condition*, Preprint arXiv:1510.00669 [math.CT], October 2015.

[38] Richard Garner, *Factorisation axioms for type theory*, Notes for a talk given at the Workshop "Identity Types – Topological and Categorical Structure" (Uppsala, November 13-14, 2006), available at `http://comp.mq.edu.au/~rgarner/Papers/Uppsala.pdf`, 2006.

[39] ———, *Understanding the small object argument*, Applied Categorical Structures **17** (2009), no. 3, 247–285.

[40] Paul G. Goerss and John F. Jardine, *Simplicial homotopy theory*, Progress in Mathematics, no. 174, Birkhäuser Basel, 1999.

[41] Georges Gonthier, *Formal proof–the four-color theorem*, Notices of the AMS **55** (2008), no. 11, 1382–1393.

[42] Georges Gonthier, Andrea Asperti, Jeremy Avigad, Yves Bertot, Cyril Cohen, François Garillot, Stéphane Le Roux, Assia Mahboubi, Russell O'Connor, Sidi Ould Biha, Ioana Pasca, Laurence Rideau, Alexey Solovyev, Enrico Tassi, and Laurent Théry, *A machine-checked proof of the odd order theorem*, ITP 2013, 4th Conference on Interactive Theorem Proving (Rennes, France) (Sandrine Blazy, Christine Paulin, and David Pichardie, eds.), Lecture Notes in Computer Science, vol. 7998, Springer, July 2013, pp. 163–179.

[43] Marco Grandis and Luca Mauri, *Cubical sets and their site*, Theory and Applications of Categories **11** (2003), no. 8, 185–211.

[44] Alexandre Grothendieck, *Pursuing stacks*, Manuscript, 1983.

[45] Michael Hedberg, *A coherence theorem for Martin-Löf's type theory*, Journal of Functional Programming **8** (1998), no. 04, 413–436.

[46] Martin Hofmann, *Extensional concepts in intensional type theory*, Ph.D. thesis, University of Edinburgh, 1995.

[47] ———, *Syntax and semantics of dependent types*, Semantics and logics of computation (A.M. Pitts and P. Dybjer, eds.), Publ. Newton Inst., vol. 14, Cambridge University Press, Cambridge, 1997, Papers from the Summer School held at the University of Cambridge, Cambridge, September 1995, pp. 79–130.

[48] Martin Hofmann and Thomas Streicher, *Lifting Grothendieck universes*, Unpublished Note.

[49] ———, *The groupoid interpretation of type theory*, Twenty-five years of constructive type theory (Venice, 1995), Oxford Logic Guides, vol. 36, Oxford University Press, New York, 1998, pp. 83–111.

[50] Simon Huber, *A model of type theory in cubical sets*, Licentiate thesis, University of Gothenburg, 2015.

[51] ———, *Canonicity for cubical type theory*, Preprint arXiv:1607.04156v1 [cs.LO], July 2016.

[52] John A. Kalman, *Lattices with involution*, Transactions of the American Mathematical Society **87** (1958), 485–491.

[53] Daniel M. Kan, *Abstract homotopy. I*, Proceedings of the National Academy of Sciences of the United States of America **41** (1955), no. 12, 1092–1096.

[54] Chris Kapulkin and Peter LeFanu Lumsdaine, *The simplicial model of univalent foundations (after Voevodsky)*, Preprint arXiv:1211.2851v4 [math.LO], November 2012.

[55] François Lamarche, *A proposal about foundations I*, Manuscript, 1991.

[56] Daniel R. Licata and Guillaume Brunerie, *A cubical approach to synthetic homotopy theory*, 30th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2015, Kyoto, Japan, July 2015, pp. 92–103.

[57] Peter LeFanu Lumsdaine, *Higher categories from type theories*, Ph.D. thesis, Carnegie Mellon University, 2010.

[58] ———, *Weak ω-categories from intensional type theory*, Logical Methods in Computer Science **6** (2010), no. 3:24, 1–19.

[59] Per Martin-Löf, *An intiutionistic theory of types: Predicative part*, Logic Colloquium '73 (H. E. Rose and J. Shepherdson, eds.), North–Holland, Amsterdam, 1975, pp. 73–118.

[60] ———, *Constructive mathematics and computer programming*, Logic, Methodology and Philosophy of Science, VI, 1982, pp. 153–175.

[61] ———, *Intuitionistic type theory*, Bibliopolis, 1984.

[62] ———, *An intuitionistic theory of types*, Twenty-five years of constructive type theory (Venice, 1995), Oxford Logic Guides, vol. 36, Oxford Univ. Press, New York, 1998, pp. 127–172.

[63] J. Peter May, *Simplicial objects in algebraic topology*, Van Nostrand Mathematical Studies, vol. 11, Van Nostrand, 1967.

[64] Guilhem Moulin, *Internalizing parametricity*, Ph.D. thesis, Chalmers University of Technology, 2016.

[65] Thomas Nikolaus, *Algebraic models for higher categories*, Indagationes Mathematicae **21** (2011), no. 1–2, 52–75.

[66] Ian Orton and Andrew M. Pitts, *Axioms for modelling cubical type theory in a topos*, 25th EACSL Annual Conference on Computer Science Logic (CSL 2016) (Dagstuhl, Germany) (Jean-Marc Talbot and Laurent Regnier, eds.), Leibniz International Proceedings in Informatics (LIPIcs), vol. 62, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2016, pp. 24:1–24:19.

[67] Erik Parmann, *Case studies in constructive mathematics*, Ph.D. thesis, University of Bergen, 2016.

[68] Christine Paulin-Mohring, *Inductive definitions in the system Coq - rules and properties*, Proceedings of the conference Typed Lambda Calculi and Applications (Marc Bezem and Jan Friso Groote, eds.), Lecture Notes in Computer Science, no. 664, 1993.

[69] Andrew M. Pitts, *An equivalent presentation of the Bezem-Coquand-Huber category of cubical sets*, Preprint arXiv:1401.7807 [cs.LO], December 2013.

[70] ———, *Nominal sets: Names and symmetry in computer science*, Cambridge Tracts in Theoretical Computer Science, vol. 57, Cambridge University Press, 2013.

[71] ———, *Nominal presentation of cubical sets models of type theory*, 20th International Conference on Types for Proofs and Programs (TYPES 2014) (Dagstuhl, Germany) (H. Herbelin, P. Letouzey, and M. Sozeau, eds.), Leibniz International Proceedings in Informatics (LIPIcs), vol. 39, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2015, pp. 202–220.

[72] Andrew Polonsky, *Extensionality of lambda-**, 20th International Conference on Types for Proofs and Programs (TYPES 2014) (Dagstuhl, Germany) (H. Herbelin, P. Letouzey, and M. Sozeau, eds.), Leibniz International Proceedings in Informatics (LIPIcs), vol. 39, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2015, pp. 221–250.

[73] ———, *Internalization of extensional equality*, Preprint arXiv:1401.1148v3 [cs.LO], January 2015.

[74] Ross Street, *Cosmoi of internal categories*, Transactions of the American Mathematical Society **258** (1980), no. 2, 271–318.

[75] Thomas Streicher, *Semantics of type theory*, Progress in Theoretical Computer Science, Birkhäuser Basel, 1991.

[76] ———, *Identity types vs. weak ω-groupoids*, Talk given at the Workshop "Identity Types – Topological and Categorical Structure" (Uppsala, November 13-14, 2006), 2006.

[77] Andrew Swan, *An algebraic weak factorisation system on 01-substitution sets: A constructive proof*, Preprint arXiv:1409.1829 [math.LO], September 2014.

[78] William W. Tait, *Intensional interpretations of functionals of finite type I*, The Journal of Symbolic Logic **32** (1967), no. 2, 198–212.

[79] The Univalent Foundations Program, *Homotopy type theory: Univalent foundations of mathematics*, `http://homotopytypetheory.org/book`, Institute for Advanced Study, 2013.

[80] Benno van den Berg and Richard Garner, *Types are weak ∞-groupoids*, Proceedings of the London Mathematical Society **102** (2011), no. 2, 370–394.

[81] ———, *Topological and simplicial models of identity types*, Transactions of the ACM on Computational Logic **13** (2012), no. 1, 3:1–3:44.

[82] Vladimir Voevodsky, *Notes on type systems*, Unpublished note available at `https://github.com/vladimirias/old_notes_on_type_systems/raw/master/old_notes_on_type%20systems.pdf` (retrieved August 3, 2016), labelled "Started September 8, 2009".

[83] ———, *Foundations of mathematics and homotopy theory*, March 2006, Talk given at the IAS, available at `https://video.ias.edu/node/68`.

[84] ———, *Notes on homotopy λ-calculus*, Unpublished note available at `https://github.com/vladimirias/2006_03_Homotopy_lambda_calculus/raw/master/homotopy_lambda_calculus_Mar_5_2006.pdf` (retrieved August 3, 2016), 2006.

[85] ———, *Univalent foundations project*, A modified version of an NSF grant application, October 2010.

[86] ———, *Univalent foundations*, Plenary lecture at WoLLIC, May 18, 2011.

[87] ———, *The equivalence axiom and univalent models of type theory. (Talk at CMU on February 4, 2010)*, Preprint arXiv:1402.5556 [math.LO], 2014.

[88] ———, *An experimental library of formalized mathematics based on the univalent foundations*, Mathematical Structures in Computer Science **25** (2015), 1278–1294.

[89] Vladimir Voevodsky, Benedikt Ahrens, Daniel Grayson, et al., *UniMath: Univalent Mathematics*, Available at `https://github.com/UniMath`.