Univalent Types, Sets and Multisets
—Investigations in dependent type theory

Håkon Robbestad Gylterud

Stockholm University

Univalent Types, Sets and Multisets

Investigations in dependent type theory

Håkon Robbestad Gylterud

ISBN 978–91–7649–654–1
ISBN 978–91–7649–655–8 (PDF version)

Printed in Sweden by US-AB, Stockholm 2017
Distributor: Department of Mathematics, Stockholm University

Sammendrag

Avhandlingen består av fire artikler i matematisk logikk og én formaliseringdel. De fire artiklene er arbeider innenfor området typeteori. De to første artiklene er arbeider internt i typeteorien, og formaliseringen av disse er skrevet i Agda – et bevissjekkingssystem basert på Martin-Löfs type teori.

Den første artikkelen omhandler *multimengder* i typeteori. Multimengder er et kjent begrep fra områder som kombinotarikk og informatikk. Kort beskrevet er multimengder samlinger av elementer hvor et element kan forekomme et vilkårlig antall ganger i samlingen. Formålet med artikkelen er å beskrive et hierarki av iterative multimengder, og utforske aksiomer for disse som ligner de man kjenner fra konstruktiv mengdelære. Homotopitypeteori og Voevodskys univalensaxiom spiller en sentral rolle, ettersom hierarkiet av multimengder bygges relativt til et univalent univers.

Den andre artikkelen tar ibruk hierarkiet av iterative multimengder fra den første artikkelen, og utvikler en modell for mengdelæren i homotopitypeteori. Dette gjøres ved å definiere mengder som de multimengder hvor hvert element forekommer høyst én gang. Vi viser at denne modellen tilfredstiller aksiomer for konstruktiv mengdelære, og at den er ekvivalent til en allerede kjent modell for mengdelæren. En av attraksjonene ved denne formuleringen er at den kan uttrykkes uten såkalte *høyere induktive typer*.

De to artikklene om multimengder og mengder er formalisert i Agda, og bevisene er sjekket ved hjelp av datamaskin. Kildekoden for formaliseringen er gjengitt, sammen med en kort diskusjon, som en separat del i denne avhandlignen.

De to siste artikklene omhandler semantikk for typeteori fra et kategoriteoretisk perspektiv. Først diskuteres en mulig kobling mellom typeteori og databaseteori, ved å konstruere en modell for typeteorien basert på simplisialkomplekser. Vi utforsker hvordan ulike typeteoretiske konsepter, slik som Σ-typer, Π-typer og univers kan oversettes via denne modellen til databaseteoretiske termer. For eksempel viser det seg at *naturlig join* er et spesialtilfelle av Π-typen i denne modellen.

Den siste artikkelen diskuterer to ulike måter å beskrive avhengighetsrelasjoner mellom termer i avhengige typesystemer. Den første måten er en variant av *kategorier med attributter*, som er en vellstudert måte å gi kategoriteoretisk semantikk til typeteori. Den andre er en videreutvikling av Makkai's såkalte enveiskategorier, til å inkludere termer i tillegg til typer.

Contents

Preface

This text has been written as a doctorate thesis in the subject of mathematical logic, and is a collection of papers. The thesis is based on research carried out during the years 2012–2016, and consists of five parts. The parts A, B, D and E are individual research papers, while Part C is a formalisation of Part A and Part B in the Agda language. Each part is equipped with an abstract and more careful introduction. We will here give a bit of context for each part.

Part A. Having studied containers for my Master Thesis at The University of Oslo, 2011, it was natural to continue to study polynomial functors and W-types in the context of Martin-Löf type theory when I arrived in Stockholm, January 2012. Through reading Egbert Rijke's master thesis, and attending the 4th Formal Topology Workshop in Ljubljana, June 2012, I became aware of what is now called Homotopy Type Theory, and the novel interpretation of the identity type as paths in a space.

In late 2013 I was studying the W-types of groupoids and their identity type when I considered the W-type, $W_{a:U} T\, a$ for a universe T. Erik Palmgren, my advisor, quickly pointed me to Aczel's 1977 paper, which uses this exact type to model set theory. Applying what homotopy theory tells us about the identity type of the universe, I arrived at the conclusions found in Part A.

Part B. The work on constructing a model of constructive set theory from the multisets of Part A started while I was visiting Carnegie Mellon University in Pittsburgh, Pennsylvania, late January and early February of 2014. At the seminar there, I presented my ideas, and Steve Awodey raised the question of how to turn this into a model of set theory. Answering this question then became the focus of Part B of this thesis.

Part C. While in Pittsburgh, I also started formalising my results on multisets in Agda. Having experimented with Agda since the very first weeks of coming to Stockholm, I was happy to find that my work on multisets was very amendable to formalisation. The work on formalising these results continued for more than a year, coming to essential completion in August 2015, after a quiet month of focused effort in the pleasant Stockholm summer. It is now collected in Part C.

Part D. The fourth part is based on previous work by co-author David I. Spivak on the connections between simplicial complexes and databases. Along with Henrik Forssell, who initiated the cooperation, we worked out the details of a model of type theory based on simplicial complexes (which form a locally cartesian closed category), and made connections back to notions in databases such as natural join. In January 2016 I presented this at Logical Foundations of Computer Science (LFCS16), and a shorter version of the article was printed in the proceedings of the conference. The full version of Part D is submitted for the post-conference special volume of Annals of Pure and Applied Logic.

Part E. During the spring term 2013 I took a course on the Theory of Operads, taught by Sergei Merkulov. Inspired by the operad approach to algebras, and Makkai's one-way categories, I wanted to study dependent type theory from a more combinatorial perspective. Some small progress on this topic was made in the following couple of years and presented in various forms at the Stockholm Logic Seminar. This work has now been collected into Part E.

Organisation of the thesis

The thesis is divided into five parts, referred to by the Latin letters A, B, C, D and E. Each part contains a number of sections, numbered 1, 2 etc. The sections are sometimes subdivided into subsections: 3.1, 3.2, etc. Definitions, lemmas, propositions and theorems are collectively numbered within each part. For instance, Lemma A:6 is followed by Definition A:7. The parts each start with an abstract and the first section of each part is an introduction. The list of references are found at the end of each part.

The mathematical notation varies slightly between the parts, reflecting that these are individual works of mathematics here collected. Hopefully, the reader will find that each part introduces its notation clearly.

Acknowledgements

The list is long of people whose discussions and encouragements have helped form and motivate the work presented here. Special thanks go to my advisor Erik Palmgren, who patiently has supported my work, and contributed his immense experience and knowledge.

Henrik Forssell and David I. Spivak, my coauthors on Part D, I would like to thank for their cooperation. Henrik has furthermore been the coadvisor of my thesis work. I would like to thank him for including me and inviting me to cooperate with him on several research projects, and for his office door always being open when I have needed someone to discuss with.

The logic group at the Mathematical Department of Stockholm University has been a really great environment to do research in. Erik and

Henrik I have already mentioned. Many thanks to them and the rest of the group: Per, Peter, Christian, Jacopo, Johan and Anna — you have been great colleagues and your own research has greatly inspired mine.

During my time at Stockholm University, I have also been fortunate enough to visit many other institutions in Europe and North America. Thanks go to all the wonderful researchers I have had the honour to meet, for their questions, comments and discussions — in particular to Steve Awodey of Carnegie Mellon University whose questions inspired what would become Part B of this thesis.

Last, but not least, I would like to thank my family for their support and help — Inna for her loving support, Yngvar for his kindness and good mood, Ingrid for long talks and proof-reading and my parents for always being there for me and from an early age allowing me to pursue my interests in mathematics. No research could have been done without such a great home support team.

—Håkon Robbestad Gylterud
Stockholm, 2017

Introduction to the thesis

Each part of the thesis has an individual section devoted to introduction. This part is intended as a quick introduction, focusing on the ideas behind each part.

Type theory

Martin-Löf's intuitionistic type theory serves as foundation of constructive mathematics. For a complete introduction we refer the reader to Nordström, Petersson, and Smith 2000. We will here give a high level overview of the aspects relevant to the articles of this thesis.

At the core of Martin-Löf's type theory are ideas such as

- *propositions as types*, sometimes referred to as the Curry—Howard correspondence,
- defining inductive structures through *introduction and elimination rules*, and
- collecting types into *universes*, in a way similar to Grothendieck universes do in set theory.

The way these three are accomplished is by having *dependent types.* A dependent type is a type which takes parameters in other (possibly themselves dependent) types. A typical first example is the type of vectors over some base type, say A. The vectors have different length and thus one can see vectors as a type $\mathrm{Vec}_A\ n$ in the context $(n : \mathbb{N})$, meaning that n for each natural number n there is a type $\mathrm{Vec}_A\ n$ of vectors of length n, of elements of A. One sometimes use the term "family of types" to denote dependent types.

Often, one can express dependent types as functions into a type of types. For instance, if Type is the type of small types, then a family of small types parameterised by a type A can be represented by a function $A \to \mathrm{Type}$.

Propositions as types

Perhaps the most appealing aspect of dependent type theory is that one does not need a separate framework for logic. Instead the type theory comes equipped with a logical framework — where types play the role of propositions, and in particular dependent types play the role of predicates. For instance, a unary predicate on a type A is simply a dependent type $P : A \to \mathrm{Type}$. A binary predicate can be seen either as a dependent type $A \times A \to \mathrm{Type}$ or, more conveniently, $A \to A \to \mathrm{Type}$ — by currying.

The beauty of representing propositions by types is that it turns out that the logical connectives can be expressed by the usual type formations, such as dependent products and sums. For instance the existential quantification of a binary predicate $P : A \to \mathrm{Type}$ is expressed by $\sum_{a:A} P\,a$. The elements of $\sum_{a:A} P\,a$ are pairs (a, p) where $a : A$ and $p : P\,a$, which is exactly a witness of the existential quantification there is $a : A$ such that $P\,a$ holds (i.e. has a witness $p : P\,a$). The table below summarises the correspondence.

Logical connective	Type formation
$\forall a : A\, P\,a$	$\prod_{a:A} P\,a$
$\exists a : A\, P\,a$	$\sum_{a:A} P\,a$
$P \to Q$	$P \to Q \equiv \prod_{p:P} Q$
$P \wedge Q$	$P \times Q \equiv \sum_{p:P} Q$
$P \vee Q$	$P + Q$
$\bot$	0, the empty type
$\top$	1, the unit type

Since a type may have more than one element, logic as presented above is called *proof-relevant* logic. The idea is that each element of the type representing a proposition represents a proof of that proposition. An added benefit of having an element of a type representing a proof of a proposition is that the element may be normalised — an important property of type theory. This means that one can often extract algorithms from proofs in type theory.

Proof-relevance is especially interesting in the case of equality. There are several ways to represent equality in type theory. One way, rooted in the ideas of Errett Bishop, is to equip the type with a separate equality predicate to form what is called a *setoid*. A second way is to represent equality by what is called *the identity type*. Given any type A, the identity type, Id_A is inductively defined as the least reflexive relation on the type.

An amazing fact, first established by Hofmann and Streicher[1] by their groupoid interpretation of type theory, is that the identity type may have more than one element. Thus, two elements of a type may be equal in more than one way. This has become the foundation of what is called *Homotopy Type Theory* — where types are interpreted as

[1]Hofmann and Streicher 1998.

a space and the identity type is the path space. For a comprehensive introduction to this field, see the book "Homotopy Type Theory"[2].

One of the deeper notions which has been brought to attention by Homotopy Type Theory is the notion of equivalence of types. We refer to the book, "Homotopy Type Theory", for the definition, but we will use the notation $A \simeq B$ to denote that A and B are equivalent types.

Introduction and elimination rules

In the previous subsection we mentioned different type constructors, such as Π-types and Σ-types. In Martin-Löf's type theory these are primitive operations on types, presented each by a set of rules. For each type there is a formation rule, none or more introduction rules, an elimination rule, and none or more computation rules. We will not display these rules here, but rather give some intuition as to what they express.

In short, the formation rules tell us how to construct types, and introduction rules how to construct elements of types. Elimination rules give sufficient conditions to carry out a construction with a free variable in the type. For instance, the elimination rule of $\mathbb{N}$ corresponds to mathematical induction by propositions as types. Computation rules tell us the result of applying the a construction specified by an elimination rule to an element constructed by an introduction rule. We refer the reader to Nordström, Petersson, and Smith 2000 for a complete description of these concepts.

Universes

In the usual formulations of dependent type theory there is one kind of types which does not have an elimination — namely the universes. The intuition behind universes is that they are "open-ended" families of types, closed under type formation rules such as Π-types and Σ-types. This means that if $A : U$ is a type in the universe[3] and $F : A \to U$ is a family of types in U, indexed by A, then $\sum_{a:A} F\,a : U$, etc.

The lack of an elimination rule has the consequence that the identity type on U is undecided. This leaves room for additional axioms specifying how to interpret the identity of the universe. The most famous

[2]Univalent Foundations Program 2013.

[3]Elements of types such as U are not themselves types, and one often speak of a decoding family $T : U \to \mathrm{Type}$, when defining a universe formally. For simplicity, we apply the syntactic convention which omits mention of this decoding family, and write $A : \mathit{Type}$ instead of $T\,A : \mathrm{Type}$

such axiom is *Voevodsky's Univalence Axiom.* It states that the identity type on the universe coincides with equivalence of types. Concretely, it states that for each $A, B : U$ the canonical map $Id_U\, A\, B \to A \simeq B$, is an equivalence of types.

Multisets

Multisets can be simply described as collections of elements where each element may occur any number of times. Examples, such as $\{1, 1, -2\}$, are abundant in mathematics, for instance as the roots of polynomials, such as $x^3 - 3x + 2$. One can even view polynomials with coefficients in natural numbers as finite multisets of finite multisets of variables. For instance $xy^2 + 3xy + x + 2$ could be represented by $\{\{x, y, y\}, \{x, y\}, \{x, y\}, \{x, y\}, \{x\}, \{\}, \{\}\}$.

Going beyond the finite case, any function gives rise to a *multiset image*, where each element in the codomain occurs the number of times the function attains this value. For instance $f : \mathbb{R} \to \mathbb{R}$, given by $f\, x := x^3 - 3x + 2$ (see Figure 1), would have an image multiset: $\operatorname{Im} f = (-\infty, 4) \cup [0, 4] \cup (0, \infty)$. Union of multisets is additive, so that the number of times x occurs in $A \cup B$ is the sum of the times x occurs in A and the times x occurs in B. Thus, $\operatorname{Im} f$ is the multiset in which each element of $(0, 4)$ occurs trice, 4 and 0 occur twice, and elements of $(-\infty, 0)$ and $(4, \infty)$ occur once. This gives a lot more information about the polynomial, compared to *the image set* — which is just $\mathbb{R}$ for any polynomial of degree 3.

A multiset may also be infinite in the sense that an element may occur infinitely main times. For instance 0 occurs countably infinitely many times in the image of the sine function, since $\sin x = 0 \;\Leftrightarrow\; \exists k \in \mathbb{Z}\; x = \pi k$.

The first article of this thesis concerns multisets. In particular *iterative multisets.* Quoting from the introduction of Part A:

> In a flat multiset, the elements are taken from some domain which may not consist of other multisets. The iterative multisets have elements which are multisets themselves, and the collection iterative multisets is generated in a well-founded manner.

The idea is to have a similar structure as in usual (iterative) set theory, where there is a domain V of sets and a binary relation $\in$ on V. This is where the idea of propositions as types enter the picture. We will have a domain M and a binary relation $\in$ on M. However, since we

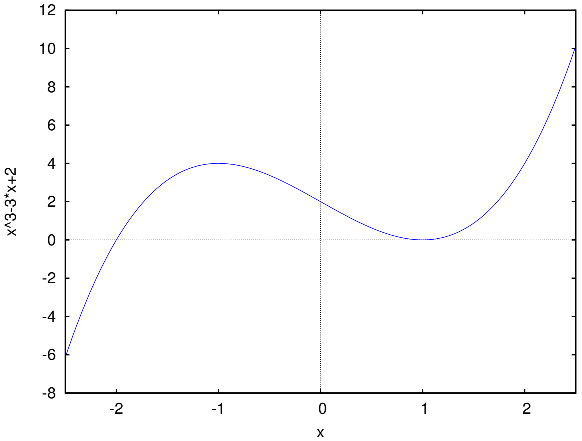

Figure 1: Plot of $f\,x := x^3 - 3x + 2$.

will be working in type theory, $x \in y$ will be a *type* for all $x, y : M$. The natural interpretation of the elements of $x \in y$ is that they represent the occurrences of x in y, and thus proof-relevance dictates that x may occur multiple times in y, hence they are multisets. We see that merely stating the signature of set theory in type theory has brought us to consider the possibility of multisets. Thus, the type M will be the type of iterative multisets M.

Letting $x \in y$ be a type is a convenient notation for multisets. For instance, stating that 0 occurs countably many times in $\operatorname{Im}\sin$ is simply asserting that $(0 \in \operatorname{Im}\sin) \simeq \mathbb{N}$.

We have already mentioned that any function gives rise to a multiset. Part A, is based on the idea that this is an adequate way to represent multisets in general, and in particular that an iterative multiset can be seen as a type $A : U$ and a function $f : A \to M$. This gives rise to an inductive definition, namely that M is the least solution to the equation $M \simeq \sum_{A:U} (A \to M)$. This is an example of a well-known inductive construction, namely W-types. In fact this is the exact type studied by Aczel 1978, in his construction of a model of set theory in type theory.

In his work, Aczel uses the setoid approach to equality. The Univalence Axiom, however, allows us to compute the identity on M. Interestingly, the identity type on M is non-trivial, with several distinct equalities even between concrete finite multisets in M. Thus, M is a groupoid, and we can see multiset theory as a kind of categorification of set theory.

Sets

The idea behind Part B is that iterative sets are merely a special class of iterative multisets, namely those in which each element occurs at most once — and such that this property is hereditary, so that each element again has the iterative set property. We define such a subtype of M, by induction, and consider how various axioms of constructive set theory apply to this model. Quoting from the introduction:

> Once the notion of a multiset is defined, it is natural to study the hereditary subtype of multisets where each element occurs at most once. These are in a certain sense the most natural representations of iterative sets from a homotopy type theory point of view. These are namely the multisets for which the elementhood relation is hereditarily, merely propositional (type level -1).
>
> In this text we explore how this type models various axioms of constructive set theory. We also show that it is equivalent to the higher inductive type outlined in the book "Homotopy Type Theory"[4].

Databases

While the first two articles of the thesis are completely situated inside type theory, Part D takes a step out and considers particular a model of type theory from a category theoretic point of view. The particular model is based on simplicial complexes, and is intended to model certain aspects of database theory. Quoting from the introduction:

> Databases being, essentially, collections of (possibly interrelated) tables of data, a foundational question is how to best represent such collections of tables mathematically in order to study their properties and ways of manipulating them. The relational model, essentially treating tables as structures of first-order relational signatures, is a simple and powerful representation. Nevertheless, areas exist in which the relational model is less adequate than in others. One familiar example is the question of how to represent partially filled out rows or missing information. Another, more fundamental perhaps, is how to relate instances of different schemas, as

[4]Univalent Foundations Program 2013.

> opposed to the relatively well understood relations between instances of the same schema. Adding to this, an increasing need to improve the ability to relate and map data structured in different ways suggests looking for alternative and supplemental ways of modelling tables, more suitable to "dynamic" settings. It seems natural, in that case, to try to model tables of different shapes as living in a single mathematical structure, facilitating their manipulation across different schemas.
>
> We investigate, here, a novel way of representing data structured in systems of tables which is based on simplicial sets and type theory rather than sets of relations and first-order logic.

The basic notions of databases are those of a database *schema* and those of an *instance of a schema*. Simply put, a schema is a description of a layout of tables: each table described by a list of *attributes*. It is essential that attributes may be shared across different tables. An instance is then an actual set of tables, filled with data, which adhere to the layout of the schema.

Given a schema and an instance, *full tuple* is a tuple of data with an entry for each attribute in the schema, such that the restriction to each table corresponds to an existing row in the instance. Below is a simple example.

Schema: $\{(A, B, C), (A, D)\}$
Instance:

A	B	C
x	a	3
y	b	7
x	b	1

A	D
x	$\top$
x	$\perp$
y	$\perp$
z	$\top$

Full tuples: $(x, a, 3, \top)$, $(x, c, 1, \top)$, $(x, a, 3, \perp)$, $(x, c, 1, \top)$ and $(y, b, 7, \perp)$.

The idea we investigate in Part D is to align these three basic notions, along with a further notion of morphism between schemas, with the basic judgements of type theory. The following table, from the article summarises this alignment:

Judgement	Interpretation
$\Gamma : \texttt{Context}$	$[\![\Gamma]\!]$ is a schema
$A : \texttt{Type}(\Gamma)$	$[\![A]\!]$ is an instance of the schema Γ
$t : \texttt{Elem}(A)$	$[\![t]\!]$ is an full tuple in the instance A
$\sigma : \Gamma \longrightarrow \Lambda$	$[\![\sigma]\!]$ is a (display) schema morphism
$\Gamma \equiv \Lambda$	$[\![\Gamma]\!]$ and $[\![\Lambda]\!]$ are equal schemas
$A \equiv B : \texttt{Type}(\Gamma)$	$[\![A]\!]$ and $[\![B]\!]$ are equal instances of $[\![\Gamma]\!]$
$t \equiv u : \texttt{Elem}(A)$	$[\![t]\!]$ and $[\![u]\!]$ are equal full tuples in $[\![A]\!]$
$\sigma \equiv \tau : \Gamma \longrightarrow \Lambda$	the morphisms $[\![\sigma]\!]$ and $[\![\tau]\!]$ are equal

References

Aczel, Peter (1978). "The Type Theoretic Interpretation of Constructive Set Theory". In: *Logic Colloquium '77*. Ed. by A. MacIntyre, L. Pacholski, and J. Paris. North–Holland, Amsterdam-New York, pp. 55–66.

Hofmann, Martin and Thomas Streicher (1998). "The groupoid interpretation of type theory". In: *Twenty-five years of constructive type theory (Venice, 1995)*. Vol. 36. Oxford Logic Guides. New York: Oxford Univ. Press, pp. 83–111.

Nordström, B., K. Petersson, and J. M. Smith (2000). *Martin-Löf's Type Theory*.

Univalent Foundations Program, The (2013). *Homotopy Type Theory: Univalent Foundations of Mathematics*. Institute for Advanced Study: homotopytypetheory.org. URL: `http://homotopytypetheory.org/book`.