Vortragsreihe zur Homotopietypentheorie

Felix Wellen

October 1, 2014

Contents

Vorwort

Dies ist der Mitschrieb zu meiner Vortragsreihe zur Homotopietypentheorie, der vor allem den Zuhörern zum Nachschlagen und Wiederholen dienen soll. Es ist sicher eine gute Idee begleitend in [Uni13] zu lesen. Außerdem verarbeite ich hier noch das während der Vorträge gelernte. Das heißt, gelegentlich gibt es hier noch zusätzliche Aussagen oder der Inhalt weicht leicht vom Vorgetragenen ab.

Grundsätzlich möchte ich stets Struktur – womit ich hier logischen Aufbau meine – zugunsten der Zugänglichkeit opfern, habe aber trotzdem vor, gelegentlich Strukturierung nachzuliefern und dabei entstehende Redundanz in Kauf zu nehmen.

Wer an irgendeiner Stelle hängt, kann mir sehr gern eine email schreiben. Für Kritik und Anregungen bin ich natürlich immer dankbar.

Wichtig: Leider taucht ganz am Ende noch die vollkommen unbewiesene Tatsache auf, dass $\mathbb{Z}$ eine Menge ist. Vielleicht liefere ich das noch nach.

1 Typentheorie

Die Typentheorie, die im Folgenden vorgestellt wird, ist eine Variante der von Per Martin-Löf erdachten und erstmals in [Mat72] auftauchenden intuitionistischen Typentheorie. Die Darstellung orientiert sich stark an den ersten Kapiteln von [Uni13].
Eine Typentheorie besteht aus *Urteilen* und *Regeln*, die festlegen, wie Urteile gefällt werden können. In unserer Typentheorie gibt es die vier Urteile:

Urteile

Urteil	Beschreibung
A ist ein Typ	auch: A ist Typ, A Typ
$a : A$	a ist ein Term des Typs A (dabei wird implizit vorausgesetzt, dass A ein Typ ist)
$x \equiv y : A$	x und y sind identische Terme des Typs A man sagt auch, dass x und y per Definition gleich sind oder dass x und y urteilsmäßig gleich sind (hier wird implizit vorausgesetzt, dass A ein Typ ist und x und y Terme des Typs A sind). Gelegentlich wird der Zusatz „$: A$“ auch weggelassen
$A \equiv B$	A und B sind identische Typen. Das Urteil kann nur gefällt werden, wenn A und B Typen sind

Die Buchstaben in den Urteilen stehen natürlich auch stellvertretend für kompliziertere Ausdrücke. Für Typen, die bereits auf syntaktischer Ebene identisch sind, gilt die urteilsmäßige Gleichheit. Insbesondere gilt für Typen A stets $A \equiv A$ und ebenso für Terme $a : A$ stets $a \equiv a : A$. Per Urteil für identisch befundene Terme oder Typen können stets ausgetauscht werden. Auf Ausführung von Details dieser Austauschvorgänge wird hier verzichtet.
Regeln erlauben uns aus bereits gefällten Urteilen neue Urteile zu schließen. In den nun folgenden Abschnitten werden einige Regeln beispielhaft herausgegriffen. Reihenfolge und Auswahl sollen dabei einen möglichst einfachen Zugang ermöglichen. Häufig werden auch zunächst Spezialfälle allgemeinerer Regeln eingeführt.

1.1 Produkte und Funktionen

Produkte in unserer Typentheorie sind den aus der Mengenlehre bekannten kartesischen Produkten sehr ähnlich. Die folgende Regel erklärt, wie Produkttypen gebildet werden können:

Wenn A ein Typ ist und B ein Typ ist, dann ist auch $A \times B$ ein Typ. $\quad (\times\mathrm{Z})$

Die beiden Urteile „A ist ein Typ“ und „B ist ein Typ“ erlauben es laut dieser Regel das Urteil „$A \times B$ ein Typ“ zu folgern. Das „Z“ im Kürzel (×Z) steht für Zusammensetzung. Wir werden noch weitere Zusammensetzungsregeln kennen lernen, deren Kürzel jeweils auch auf „Z“ enden. Die nächste Regel ist ein sogenannter *Konstruktor* und erlaubt uns in diesem Fall Terme des soeben eingeführten *Produkttyps* $A \times B$ zu konstruieren:

$$\text{Sind Terme } a{:}A \text{ und } b{:}B \text{ gegeben, dann gibt es den Term } (a, b){:}A \times B. \tag{×K}$$

Genauso wie bei (×Z) erlaubt die Konstruktionsregel (×K) aus zwei Urteilen ein Drittes zu folgern.

Bemerkung 1.1.1
Wenn wir uns soweit herablassen einen Moment so zu tun als wären unsere Typen Mengen und ihre Terme Elemente, dann wüssten wir nun, dass in dem Produkttyp $A \times B$ alle Paare (a, b) enthalten sind, die auch im kartesischen Produkt der beiden Typen – als Mengen aufgefasst – enthalten wären. Wir wissen an diesem Punkt allerdings noch nichts darüber, ob es nicht vielleicht doch noch weitere Terme in $A \times B$ gibt.

Vor der nächsten Regel für den Produkttyp müssen wir uns mit Funktionstypen beschäftigen. Deren Regeln wurden leicht zurückgestellt, da der Konstruktor etwas komplizierter als der des Produkttyps ist. Die Zusammensetzungsregel unterscheidet sich allerdings lediglich im Symbol:

$$\text{Wenn } A \text{ ein Typ ist und } B \text{ ein Typ ist, dann ist auch } A \to B \text{ ein Typ.} \tag{→Z}$$

Die Konstruktionsregel hat als Voraussetzung nicht eine bloße Liste von Urteilen, sondern fordert die Ableitbarkeit des Urteils $b(a){:}B$ aus dem Urteil $a{:}A$. Dabei soll das Anhängsel (a) am Term $b(a)$ lediglich bedeuten, dass bei der Konstruktion des Terms $b(a){:}B$ der Term $a{:}A$ möglicherweise verwendet wurde, also $b(a){:}B$ von $a{:}A$ abhängt. Die folgende Regel erlaubt es nun, aus einem von $a{:}A$ abhängigen „Funktionsterm“ $b(a){:}B$ eine Funktion von A nach B zu konstruieren:

$$\text{Gilt } b(a){:}B \text{ für } a{:}A\text{, dann gibt es den Term } a \mapsto b(a){:}A \to B. \tag{→K}$$

Das „a“ im Urteil $a{:}A$ kann als Variable verstanden werden, denn „a“ steht hier wirklich für einen einzelnen Buchstaben – A und alle weiteren Buchstaben jedoch nicht notwendigerweise! Das ändert allerdings nichts daran, dass es sich bei $b(a)$ um nichts anderes handelt als einen Term des Typs B – selbst wenn a wie eine Variable darin auftaucht.

Beispiel 1.1.2

(a) Für $a{:}A$ ist $(a, a){:}A \times A$ laut (×K). Also gibt es eine Funktion, d.h. einen Term $a \mapsto (a, a){:}A \to (A \times A)$. Man beachte, dass diese Funktion immer existiert – sogar wenn es keinen Term des Typs A gibt!

(b) Wenn wir wieder eine – formal hier soweit vollkommen unbegründete – Analogie zur mengenbasierten Mathematik verwenden, so könnte etwa für einen beliebigen „Term“ $x{:}\mathbb{R}_{\geq 0}$ gefolgert werden, dass der Term $\sqrt{x}$ den Typ $\mathbb{R}$ hat, also $\sqrt{x}{:}\mathbb{R}$ gilt.

Nach der Regel (→K) gäbe es nun eine Funktion $x \mapsto \sqrt{x} : \mathbb{R}_{\geq 0} \to \mathbb{R}$. Wenn $x : \mathbb{R}$ vorausgesetzt gewesen wäre, hätte $\sqrt{x} : \mathbb{R}$ nicht gefolgert werden können und die Regel (→K) wäre nicht anwendbar gewesen.

Nun wenden wir uns für einen Moment wieder dem Produkt zu. Um eine Funktion $(A \times B) \to C$ zu konstruieren, reicht es nach Allem was wir soweit wissen nicht, einen Term $c(a, b) : C$ für jedes $(a, b) : A \times B$ anzugeben – wir können (→K) nur für ein allgemeines $x : A \times B$ anwenden. Dass es doch schon genug ist, eine Funktionterm nur für Paare $(a, b) : A \times B$ anzugeben, ist die wesentliche Aussage der folgenden Regel:

$$\text{Ist } c(a, b) : C \text{ für } a : A \text{ und } b : B, \text{ dann gibt es eine Funktion } f : (A \times B) \to C \text{ mit } f((a_0, b_0)) \equiv c(a_0, b_0) \text{ für } a_0 : A, b_0 : B. \tag{$\times$R}$$

Genauso wie bei (→K) soll das Urteil $c(a, b) : C$ unter der Voraussetzung der Urteile $a : A$ und $b : B$ gefolgert werden können und „a“ und „b“ sind wieder als einzelne Buchstaben und nicht als Platzhalter für kompliziertere Ausdrücke zu verstehen. Das „R“ im Kürzel dieser Regel steht für Rekursion – wie das zu eventuell bekannten Formen von Rekursion passt, wird später bei anderen Typen klarer zu erkennen sein. Um in Zukunft weniger Klammern zu benötigen, verschaffen wir uns mit der folgenden Konvention Notationserleichterung:

Konvention
Zusammensetzung mit „$\times$“ bindet stärker als Zusammensetzung mit „$\to$“. Daraus ergibt sich, dass „$(A \times B) \to C$“ auch als „$A \times B \to C$“ geschrieben werden kann. Weiter wollen wir für Funktionen $g : A \times B \to C$ gelegentlich statt $g((a, b))$ auch $g(a, b)$ schreiben, wenn keine Verwechslungsgefahr besteht.
Weiter wollen wir von nun an gelegentlich etwas lockerer mit Urteilen umgehen und etwa ab und zu auch mal den Zusatz „$: A$“ in einem Urteil $a : A$ weglassen.

Das folgende Beispiel schafft vielleicht den richtigen Eindruck, wie diese Regel (×R) benutzt werden kann.

Beispiel 1.1.3
Wird in (×R) für C einfach der Typ A gewählt und für $c(a, b) : C$ der Term $a : A$, dann erfüllt die Ergebnisfunktion $f : A \times B \to C$ folgende Gleichung: $f(a_0, b_0) \equiv a_0$ für $a_0 : A, b_0 : B$. Wir wollen diese Funktion f mit $\pi_{1,A}$ bezeichnen. Dabei lassen wir gelegentlich die „1“ oder das „A“ weg, wenn uns danach ist.
Analog kann eine Funktion $\pi_{2,B} : A \times B \to C$ konstruiert werden, sodass $\pi_{2,B}(a_0, b_0) \equiv b_0$ für $a_0 : A, b_0 : B$ gilt.

Bemerkung 1.1.4
Mittels der beiden Funktionen π_A und π_B aus dem Beispiel kann man alle Funktionen des Typs $A \times B \to C$ konstruieren, die man auch mit der Rekursionsregel konstruieren kann. Dazu wird es später eine konkrete Aufgabenstellung geben.

Um mehr mit Funktionen machen zu können, brauchen wir noch Regeln, die uns erlauben mit Funktionen zu „rechnen“. Für Funktionen gibt es die folgenden drei „Rechenregeln“. Die erste ermöglicht das Einsetzen von Werten:

$$\text{Für } a : A \text{ und } f : A \to B \text{ gibt es einen Term } f(a) : B. \tag{$\to$B1}$$

Wenn die Funktion über einen bekannten Term gegeben ist, kann diese bei einem Wert berechnet werden:

$$\text{Für } a_0 : A \text{ und } a \mapsto b(a) : A \to B \text{ gilt } (a \mapsto b(a))(a_0) \equiv b(a_0) : B. \tag{$\to$B2}$$

Hier bezeichnet $b(a_0)$ den Term, der aus $b(a)$ hervorgeht, wenn alle Vorkommen von a durch a_0 ersetzt werden. Es mag vielleicht auf den ersten Blick überraschen, dass die folgende Regel reichlich Anwendung findet:

$$\text{Für } f : A \to B \text{ gilt } x \mapsto f(x) \equiv f : A \to B. \tag{$\to$B3}$$

Aus diesen Regeln können wir sofort ein technisches Hilfslemma folgern:

Lemma 1.1.5
Sind $b(x), b(y) : B$ für $x, y : A$ Terme, die bis auf Umbenennung der Variablen gleich sind, also $b(x)$ aus $b(y)$ durch ersetzen aller Vorkommen von y durch x hervorgeht, dann gilt $x \mapsto b(x) \equiv y \mapsto b(y)$.

Beweis Nach ($\to$B2) gilt $x \mapsto b(x) \equiv x \mapsto (y \mapsto b(y))(x)$.
Und nach ($\to$B3) gilt $x \mapsto (y \mapsto b(y))(x) \equiv y \mapsto b(y)$. Also gilt die Behauptung.

Nun sind genug Regeln beisammen um eine halbwegs interessante Definition zu wagen.

Definition 1.1.6

(a) Für einen beliebigen Typ A sei $1_A :\equiv x \mapsto x : A \to A$ die *Identität* auf A.

(b) Für Typen A, B und C sei eine Funktion

$$\_ \circ \_ : (A \to B) \times (B \to C) \to (A \to C)$$

gegeben durch $a \mapsto g(f(a))$ für $f : A \to B$ und $g : B \to C$. Es gilt:

$$g \circ f :\equiv \_ \circ \_(f, g) \equiv a \mapsto g(f(a))$$

Und $g \circ f$ heißt *Komposition* von g und f.

Man beachte, dass A keine Terme besitzen muss, damit die Funktion 1_A existiert.

Lemma 1.1.7

(a) Für einen Typ A ist die Funktion 1_A stets links- und rechtsneutral, das heißt für $g : B \to A$ und $f : A \to C$ gilt:

$$1_A \circ g \equiv g \text{ und } f \circ 1_A \equiv f$$

(b) Die Komposition $\circ$ ist assoziativ, das heißt für $f{:}A \to B, g{:}B \to C$ und $h{:}C \to D$ gilt:
$$h \circ (g \circ f) \equiv (h \circ g) \circ f$$

Beweis (a) Wir wollen das Urteil $1_A \circ g \equiv g$ folgern, wozu wir eine Kette von urteilsmäßigen Gleichheiten herstellen, die mit der linken Seite der zu folgernden Gleichheit beginnt und mit der rechten Seite endet. Wir lösen zunächst die Definitionen von 1_A und $\circ$ in $1_A \circ g$ auf und erhalten $b \mapsto (x \mapsto x)(g(b))$. Auf diesen Term können wir die Rechenregel ($\to$B2) anwenden, um $(x \mapsto x)$ verschwinden zu lassen. Nach ($\to$B3) gilt nun: $b \mapsto g(b) \equiv g$, womit die erste urteilsmäßige Gleichheit gezeigt ist. Da die zweite auf exakt dem selben Weg gezeigt werden kann, wird hier auf einen Beweis verzichtet.

(b) Wir beginnen wieder mit der linken Seite $h \circ (g \circ f)$ und verwenden die Definition von $\circ$. Damit erhalten wir zunächst $h \circ (a \mapsto g(f(a)))$ und schließlich $a_0 \mapsto h((a \mapsto g(f(a)))(a_0))$. Auf den letzten Term können wir die Regel ($\to$B2) anwenden und erhalten $a_0 \mapsto h(g(f(a_0)))$. Nun schauen wir uns die rechte Seite $(h \circ g) \circ f$ an und verfahren ähnlich wie mit der linken um zunächst $(b \mapsto h(g(b))) \circ f$, dann $a \mapsto (b \mapsto h(g(b)))(f(a))$ und schließlich, wieder mit ($\to$B2), $a \mapsto h(g(f(a)))$ zu erhlaten. Damit sind rechte und linkte Seite bis auf Umbenennung von Variablen gleich, womit die Behauptung gezeigt ist.

Es gibt noch eine andere, kompaktere Möglichkeit, Rekursionsregeln aufzuschreiben. Dazu möchten wir allerdings zuerst noch unsere syntaktischen Freiheiten durch die folgende Konvention erweitern.

Konvention

(a) Es gilt $A \to B \to C \equiv A \to (B \to C)$. Bei längeren Ketten wählen wir in übereinstimmung damit eine „Rechtsklammerung“: $A \to B \to C \to D \equiv A \to (B \to (C \to D))$

(b) Für Funktionen $g\colon A \to B \to C$ schreiben wir gelegentlich auch $g(a, b)$ und $g(a)(b)$ statt $(g(a))(b)$.

Mit den neuen Konventionen können wir die Rekursionsregel für das Produkt ($\times$R) folgendermaßen aufschreiben:

$$\begin{aligned}&\text{Es gibt einen Term } r_{A\times B}{:}(A \to B \to C) \to (A \times B \to C) \text{ mit}\\ &r_{A\times B}(g)(a, b) \equiv g(a, b) \text{ für } g{:}A \to B \to C,\ a{:}A \text{ und } b{:}B.\end{aligned} \tag{$\times R'$}$$

Die Konvention, $g(a, b)$ anstelle von $g(a)(b)$ zu schreiben, und damit Funktionen $g\colon A \to B \to C$ mit Funktionen $g\colon A \times B \to C$ zu identifizieren, sei dem davon überraschten Leser durch die folgende Bemerkung näher gebracht:

Bemerkung 1.1.8

(a) Die Funktion $r_{A\times B}{:}(A \to B \to C) \to (A\times B \to C)$ hat eine „Inverse“. Das soll hier heißen, dass es ein $c{:}(A\times B \to C) \to (A \to B \to C)$ gibt, mit $c\circ r_{A\times B} \equiv 1_{A\to B\to C}$.

(b) Für $r_{A\times B} \circ c$ gilt Ähnliches, was wir hier aber noch nicht ausdrücken können.

Beweis Sei $c \equiv g \mapsto [a \mapsto (b \mapsto g(a,b))]$. Für $f : A \to B \to C$ gilt

$$\begin{aligned}(c \circ r_{A\times B})(f) &\equiv c(r_{A\times B}(f)) \\ &\equiv a \mapsto (b \mapsto r_{A\times B}(f)(a,b)) \\ &\equiv a \mapsto (b \mapsto f(a)(b)) \\ &\equiv a \mapsto f(a) \\ &\equiv f\end{aligned}$$

1.2 0,1,2 und Koprodukte

Nun wollen wir uns noch vier weitere Typen anschauen, die ins Regelnschema des Produkttyps passen. Der Typ 0 kann, wenn man Typen und Aussagen identifiziert, als „Falsch" aufgefasst werden. Neu ist hier, wie auch bei 1 und 2, dass es für die Zusammensetzungsregel keine Parametertypen gibt und die Liste der Konstruktoren leer ist.

Es gibt den Typ 0. (0Z)
- (Kein Konstruktor)
Für einen Typ A gibt es eine Funktion $f : 0 \to A$ (0R)

Die Regeln des Typs 1 sind sehr ähnlich:

Es gibt den Typ 1. (1Z)
$* : 1$ (1K)
Für einen Typ A und $a : A$ gibt es eine Funktion $f : 1 \to A$ mit $f(*) \equiv a$ (1R)

Mit dem Typ 2 sehen wir zum ersten Mal einen Typ mit mehr als einem Konstruktor:

Es gibt den Typ 2. (2Z)
$*_1 : 2$ (2K1)
$*_2 : 2$ (2K2)
Für einen Typ A und $a_0, a_1 : A$ gibt es eine Funktion $f : 2 \to A$ mit $f(*_1) \equiv a_1$ und $f(*_2) \equiv a_2$ (2R)

Natürlich ließe sich diese Liste beliebig weit fortsetzen – wir werden aber vor allem diese drei Typen einsetzten und später leichter sehen können, wie die gesamte Liste definiert werden kann.
Jetzt wollen wir uns noch das Koprodukt anschauen. Wie schon bei Produkt und Funktionstyp, wird das Koprodukt aus zwei gegebenen Typen zusammengesetzt:

Sind A und B Typen, dann ist $A + B$ ein Typ. (+Z)

Weiter hat das Koprodukt – wie der oben vorgestellte Typ 2 – zwei Konstruktoren:

Ist $a:A$, dann gibt es einen Term $\iota_{1,A}(a):A+B$. (+K1)

Ist $b:B$, dann gibt es einen Term $\iota_{2,B}(b):A+B$. (+K2)

Indizes von ι wollen wir natürlich weglassen, wenn keine Verwechslungsgefahr besteht. An uns soweit bekannten Regeltypen bleibt noch die Rekursion:

Ist C ein Typ und $c_1(a):C$ für $a:A$ und $c_2(b):C$ für $b:B$,
dann gibt es eine Funktion $f:(A+B) \to C$ mit
$f(\iota_1(a_0)) \equiv c_1(a_0)$ für $a_0:A$ und $f(\iota_2(b_0)) \equiv c_2(b_0)$ für $b_0:B$. (+R)

Lemma 1.2.1
Es gibt einen Term

$$d\colon (A+B \to C) \to (A \to C) \times (B \to C)$$

Beweis $d :\equiv f \mapsto (a \mapsto f(\iota_A(a)), b \mapsto f(\iota_B(b)))$

Wenn „$\to$“ als Implikation verstanden wird, entspricht folgender Definition einer eventuell bekannten Möglichkeit, logische Negation mittels Implikation zu definieren:

Definition 1.2.2
Für Typen A sei $\neg A :\equiv A \to 0$.

Konvention
„$\neg$“ bindet stärker als „$\times$“, „$+$“ und „$\to$“.

Da wir nun alle Typen kennen gelernt haben, die man für Aussagenlogik braucht, wollen wir eine Übersicht betrachten, wie unsere Typen als Aussagen interpretiert werden können:

Korrespondenz zwischen Aussagen und Typen

Typ	Aussage
0	Falsch
1	Wahr
$\neg A$	nicht A
$A \to B$	Aus A folgt B
$A \times B$	A und B
$A + B$	A oder B

Das folgende Korollar aus 1.2.1 ist ein Teil der De Morganschen Regeln:

Korollar 1.2.3
$\neg(A+B) \to \neg A \times \neg B$

Aufgabe 1.2.4
Es gibt in 1.2.1 auch eine Funktion in die andere Richtung. Finde einen solchen Term

$$d' : (A \to C) \times (B \to C) \to (A + B \to C)$$

Aufgabe 1.2.5
Sei A ein Typ. Konstruiere Terme für die folgenden Typen:

(a) $A \to \neg\neg A$

(b) $\neg\neg\neg A \to \neg A$

(c) $\neg\neg(\neg A + A)$

1.3 Universen und abhängige Typen

Motivation

Ein Universum soll so etwas wie ein Typ U sein, sodass jeder Typ ein Term von U ist. Das Tolle daran ist unter anderem, dass man mit einem Universum die bereits vorhandene Typentheorie verwenden kann um Aussagen über die Gesamtheit aller Typen zu machen ohne dafür eine Metatheorie erfinden zu müssen. Leider würde eine Regel die für jeden Typ A das Urteil $A : U$ fällen lässt unsere Typentheorie inkonsistent machen, wenn U auch ein Typ ist. Per Martin Löf hatte ähnliches in einer frühen Version seiner Typentheorie gefordert – die Inkonsistenz wurde von Girard [Gir72] gezeigt.

Möglichkeit trotzdem „ein“ Universum zu haben

Zunächst fordern wir nicht, dass es ein einziges Universum gibt, sondern gleiche eine ganze Hirarchie $U_0 : U_1 : U_2 : \dots$. Es soll also für jede Natürliche Zahl[1] i einen Typen U_i geben, sodass jeweils das Urteil $U_i : U_{i+1}$ gilt. Weiter soll jedes U_i unter Zusammensetzungen abgeschlossen sein[2]. Und jeder Typ soll Term eines U_i sein.

Konvention
Die Indizes i der U_i werden weggelassen. Das klappt gut, weil man selten mehr als ein Universum auf einmal braucht. Von nun an verwenden wir also ein *Universum* U und sollten dabei darauf achten, dass es im Zweifelsfall möglich ist, die weggelassenen Indizes der Universen anzugeben.

Definition 1.3.1
Sei A ein Typ. Ein *abhängiger Typ* über A ist eine Funktion $P : A \to U$.

[1] Ein technisches Detail: Gemeint sind natürliche Zahlen auf Metaebene. Der Index ist nicht Term des Typs der natürlichen Zahlen, der später noch eingeführt wird.

[2] Das wird später direkt in Zusammensetzungsregeln gefordert und nicht mehr eine Forderung an unsere Universen sein.

1.4 Abhängige Produkte

Das *abhängige Produkt* oder der Typ der *abhängigen Funktionen* hat die folgende Zusammensetzungsregel:

$$\text{Für einen abhänigen Typen } P\colon A \to U \text{ gibt es den Typ } \prod_{a\,:\,A} P(a) \tag{$\prod$ Z}$$

Abhängige Funktionen sind eine Verallgemeinerung der Funktionen – im Sinn von Termen von $A \to B$. Was sich ändert, ist, dass gelegentlich äbhängige Typen auftauchen, wo sonst ein fester Typ stand, etwa in der folgenden Konstruktionsregel:

$$\text{Sei } p(a)\colon P(a) \text{ für } a\colon A, \text{ dann gibt es den Term } a \mapsto p(a)\colon \prod_{a\,:\,A} P(a) \tag{$\prod$ K}$$

Die drei Rechenregeln, die wir bereits bei den Funktionen gesehen haben, sollen auch für die abhängigen Funktionen gelten, wobei auch hier die einzige Änderung ist, dass feste Typen durch abhängige ersetzt wurden.

$$\text{Für } a : A \text{ und } f : \prod_{a:A} P(a), \text{ dann gibt es den Term } f(a) : P(a). \tag{$\prod$B1}$$

$$\text{Für } a_0 : A \text{ und } a \mapsto p(a) : \prod_{a:A} P(a) \text{ gilt } (a \mapsto p(a))(a_0) \equiv p(a_0) : P(a_0). \tag{$\prod$B2}$$

$$\text{Für } f : \prod_{a:A} P(a) \text{ gilt } x \mapsto f(x) \equiv f. \tag{$\prod$B3}$$

Die Vielfätigkeit dieses Typs ist Gegenstand der folgenden Bemerkung:

Bemerkung 1.4.1

(a) Auf Seite der Logik entspricht der Typ $\prod_{a:A} P(a)$ der Aussage „Für alle $a : A$ gilt $P(a)$“.

(b) Es gibt die alternative Schreibweise $(a : A) \to P(a)$ für den Typ $\prod_{a:A} P(a)$.

(c) Wir hätten den Typ $A \to B$ auch als $\prod_{a:A} P(a)$ mit $P(a) \equiv B$ definieren können.

(d) Das Produkt $A \times B$ hätte auch als $\prod_{x:2} P(x)$ mit $P :\equiv r_{2,U}(A)(B)$ definiert werden können.

1.5 Gleichheit und Induktion

In diesem Abschnitt wenden wir uns nun endlich den Regeln für die Gleichheit „=“ zu. Diese ist unten zunächst in Anführungszeichen in einer hoffentlich leichter verdaulichen Form und schließlich in der endgültigen Fassung aufgeführt.

„Für einen Typ A und $x, y : A$ gibt es den Typ $x =_A y$.“

$$\text{Für einen Typ } A \text{ gibt es den abhängigen Typ } \_ =_A \_\colon A \times A \to U. \tag{=Z}$$

Die erste Version der Zusammensetzungsregel erlaubt es uns zwar auch den Typ $\_ =_A \_ : A \times A \to U$ zu bilden, doch lässt sich der zweiten Version leichter eine Induktionsregel entlocken, wie wir später in diesem Abschnitt noch sehen werden. Das Symbol $\_ =_A \_$ ist als zweistellige Funktion zu verstehen, das Bild von (x, y) wird mit $x =_A y$ bezeichnet. Die einzige Konstruktionsregel ist die Reflexivität, wobei auch hier wieder zuerst eine eventuell bei erstem Kontakt einleuchtendere Version aufgeführt ist:

„Für einen Typ A und $a : A$ gibt es den Term $r_a : a =_A a$."

$$\text{Für einen Typ } A \text{ gibt es den Term } r : \prod_{a:A} a =_A a. \tag{=K}$$

Nun wollen wir uns kurz von den Regeln der Gleichheit abwenden um uns Induktionsregeln bereits bekannter Typen anzuschauen. Induktionsregeln erlauben uns eine Aussage etwa über die Terme des Koprodukts $A + B$ – das heißt einen abhängigen Typen $P : A+B \to U$ – für alle $x : A+B$ zu zeigen. Die Eingangsdaten der Induktionsregel sind Nachweise, dass P auf den beiden Typen A und B gilt, also Terme $p_A : \prod_{a:A} P(\iota_A(a))$ und $p_B : \prod_{b:B} P(\iota_B(b))$. Sind solche Terme p_A und p_B gegeben, dann gibt es laut der Induktionsregel einen Term $i_+(p_A)(p_B) : \prod_{x:A+B} P(x)$, der p_A und p_B fortsetzt.

$$\text{Für } P : A + B \to U \text{ gibt es } i_+ : \left(\prod_{a:A} P(\iota_A(a))\right) \to \left(\prod_{b:B} P(\iota_B(b))\right) \to \prod_{x:A+B} P(x)$$
$$\text{mit } i_+(p_A)(p_B)(\iota_A(a)) \equiv p_A(a) \text{ und } i_+(p_A)(p_B)(\iota_B(b)) \equiv p_B(b) \tag{+I}$$

Ähnlich wie bei den Rekursionsregeln geht es also darum, das was man braucht (Beweis einer Aussage, Konstruktion einer Funktion) auf den „Einzelteilen" des Typs anzugeben. Und was diese „Einzelteilen" sind, sagen uns die Konstruktoren des Typs. Sehr ähnlich verhält es sich mit 0, 1 und 2:

$$\text{Für } P : 0 \to U \text{ gibt es } i_0 : \prod_{x:0} P(x). \tag{0I}$$

$$\text{Für } P : 1 \to U \text{ und } p_* : P(*) \text{ gibt es } i_1(p_*) : \prod_{x:1} P(x) \text{ mit } i_1(p_*)(*) \equiv p_*. \tag{1I}$$

$$\text{Für } P : 2 \to U,\ p_1 : P(*_1) \text{ und } p_2 : P(*_2) \text{ gibt es } i_2(p_1, p_2) : \prod_{x:2} P(x)$$
$$\text{mit } i_2(p_1, p_2)(*_1) \equiv p_1 \text{ und } i_2(p_1, p_2)(*_2) \equiv p_2. \tag{2I}$$

Gut ins Muster passt auch die Induktionsregel für „$A \times B$". Hier ist allerdings neu, dass wir abhängige Funktionen mit mehreren Argumenten benötigen. Den Typ solcher Funktionen, können wir, genauso wie bei normalen Funktionen durch Schachtelung ausdrücken: $\prod_{a:A} \left(\prod_{b:B} P(a, b)\right)$ ist der Typ der abhängigen Funktionen f mit $f(a)(b) : P(a, b)$. Um besser mit solchen Schachtelungen und überhaupt mit abhängigen Produkten umgehen zu können, erleichtern wir uns noch etwas die Notation:

Konvention
Genauso wie beim Funktionstyp wird implizit eine Rechtsklammerung angenommen, das soll heißen, dass $\prod_{a:A} \prod_{b:B} P(a, b)$ als $\prod_{a:A} \left(\prod_{b:B} P(a, b)\right)$ zu verstehen ist. Mehrere

Argumente des seleben Typs wollen wir unter einem Produktzeichen zusammenfassen: $\prod_{a_1,a_2:A} P(a_1,a_2) :\equiv \prod_{a_1:A}\prod_{a_2:A} P(a_1,a_2)$. Außerdem sollen Produkte soweit wie möglich nach rechts reichen, das heißt, dass etwa $\prod_{a:A} P(a) \to Q(a)$ als $\prod_{a:A}(P(a) \to Q(a))$ zu lesen ist.

Die Produktinduktion lässt sich nun folgendermaßen beschreiben:

$$\text{Für } P : A \times B \to U \text{ und } p : \prod_{a:A}\prod_{b:B} P((a,b)) \text{ gibt es } i_\times(p) : \prod_{x:A\times B} P(x)$$
$$\text{mit } i_\times(p)((a,b)) \equiv p(a)(b). \tag{×I}$$

An dieser Stelle möchten wir nun endlich mit unseren Induktionsregeln etwas beweisen.

Beispiel 1.5.1

(a) Alle Terme von 1 sind gleich $*$, das heißt, der Typ $\prod_{x:1} x =_1 *$ hat einen Term. Nach der Induktionsregel (1I) brauchen wir dazu einen Term in $* =_1 *$. Ein solcher Term ist aber durch Reflexivität gegeben: $r(*) : * =_1 *$, also gilt $i_1(r(*)) : \prod_{x:1} x =_1 *$.

(b) Nun wollen wir noch sehen, dass jeder Term von 2 gleich $*_1$ oder $*_2$ ist. Das belegt der Term $i_2(\iota_{x=*_1}(r(*_1)), \iota_{x=*_2}(r(*_2))) : \prod_{x:2}(x =_2 *_1) + (x =_2 *_2)$.

Aufgabe 1.5.2

(a) Man konstruiere einen Term von $\prod_{x:A\times B} x = (\pi_A(x), \pi_B(x))$.

(b) Erstaunlicherweise[3] gilt, dass eine Aussage, die für einen Term zutrifft, auch für jeden gleichen Term gilt. Finde für $P : A \to U$ und $\gamma : x = y$ einen Term $\mathcal{T}_\gamma : P(x) \to P(y)$ mit $\mathcal{T}_{r_x} \equiv 1_{P(x)}$.

Schwieriger ist die Induktionsregel für den Gleichheitstyp $x =_A y$, allerdings ist das wohl schon das größte Hindernis im Typentheorieteil der Vortragsreihe. Die Regel sagt uns, dass es reicht eine Aussage $P : \prod_{x,y:A} x =_A y \to U$, die auf dem Typ $\_ =_A \_$ gelten soll, für die durch Reflexivität gegebenen Gleichheiten zu zeigen, also einen Term von $\prod_{a:A} P(a,a,r(a))$ anzugeben. Dazu sollen auch wieder entsprechende Identitäten gelten. Formal:

$$\text{Für } P : \prod_{x,y:A} x =_A y \to U \text{ und } p : \prod_{a:A} P(a,a,r_a) \text{ gibt es } i_=(p) : \prod_{x,y:A}\prod_{\gamma:x=_A y} P(x,y,\gamma)$$
$$\text{mit } i_=(p)(a,a,r_a) \equiv p(a). \tag{=I}$$

und nochmal knapper:

$$\text{Für } P : \prod_{x,y:A} x =_A y \to U \text{ gibt es } i_= : \left(\prod_{a:A} P(a,a,r_a)\right) \to \prod_{x,y:A}\prod_{\gamma:x=_A y} \to P(x,y,\gamma)$$
$$\text{mit } i_=(p)(a,a,r_a) \equiv p(a). \tag{=I}$$

[3] An dieser Stelle mag das den Leser vielleicht noch nicht erstaunen. Später wird sich noch zeigen, dass äquivalente Typen auch gleich sind – womit die Aussage der Aufgabe deutlich an tragweite gewinnt.

Damit können wir sofort die Symmetrie von „=“ zeigen und gleichzeitig den Term, der die Aussage beweist, für eine wichtige Definition verwerten.

Definition + Bemerkung 1.5.3
Aus $x =_A y$ folgt $y =_A x$, genauer gibt es einen Term $i : (x = y) \to (y = x)$ mit $i(r_x) \equiv r_x$. Für $\gamma : x = y$ schreiben wir im Folgenden γ^{-1} statt $i(\gamma)$ und nennen γ^{-1} das *Inverse* von γ.

Beweis Wir wenden (=I) auf $\prod_{x,y:A} \prod_{\gamma:x=_A y} y =_A x$ an und finden mit der Reflexivität r den nötigen Term von $\prod_{a:A} a = a$.

Der Beweis der Transitivität lässt sich ebenfalls für die Definition einer Abbildung verwenden, zwei Gleichheiten zu einer dritten kombinieren. Später werden wir noch sehen, dass Gleichheiten zwischen Termen eines Typs als Pfade zwischen Punkten in einem Raum verstanden werden können. Die unten definierte Abbildung wird dem aneinanderhängen solcher Pfade entsprechen.

Definition + Bemerkung 1.5.4
Seien $x, y, z : A$ und $\gamma : x = y$ und $\gamma' : y = z$ gegeben, so gibt es $\gamma \bullet \gamma' : x = z$, sodass $r_x \bullet r_x \equiv r_x$ im Fall $x \equiv y \equiv z$ und $\gamma \equiv \gamma' \equiv r_x$ gilt.

Beweis Wir zeigen das gleichzeitig für alle x, y, z in A. Zunächst sollten wir ein geeignetes P finden. Es bietet sich an, einfach mit zwei Punkten in A und einem Pfad γ dazwischen anzufangen und anschließend erst die Abhängigkeit nach einem dritten Punkt und einem zweiten Pfad einzubauen:

$$\prod_{x,y:A} \prod_{\gamma:x=y} \underbrace{\prod_{z:A} \prod_{\gamma':y=z} x = z}_{:\equiv P(x,y,\gamma)}$$

Um (=I) anwenden zu können, ist also ein Term von

$$\prod_{a:A} P(a, a, r_a) \equiv \prod_{a:A} \prod_{a:A} a = a$$

zu konstruieren. Wir wählen für diesen Term die Abbildung, die konstant die Identität ist: $a \mapsto a \mapsto 1_{a=a}$. Damit gilt

$$i_=(a \mapsto a \mapsto 1_{a=a}) : \prod_{x,y:A} \prod_{\gamma:x=y} \prod_{z:A} \prod_{\gamma':y=z} x = z$$

und die gewünschte Abbildung $\_ \bullet \_ : x = y \to y = z \to x = z$ ist durch $\gamma \mapsto \gamma' \mapsto i_=(a \mapsto a \mapsto 1_{a=a})(x, y, \gamma, z, \gamma')$ gegeben. Außerdem gilt mit dieser Definition die urteilsmäßige Gleichheit $r_a \bullet \_ \equiv 1_{a=z}$ also insbesondere auch die geforderte urteilsmäßige Gleichheit.

Wir wollen nun ein paar einfache Aussagen über „$\bullet$“ und „$^{-1}$“ machen, die mit einer Interpretation der Gleichheiten als Pfade und Gleichheiten zwischen Pfaden als Homotopien vereinbar ist. Dabei stellen wir jeweils eine entsprechende urteilsmäßige Gleichheit auf, auch wenn diese direkt aus bereits bekannten folgen.

Bemerkung 1.5.5
Seien $\gamma : x = y$, $\gamma' : y = z$ und $\gamma'' : z = w$.

(a) $(\gamma \bullet \gamma') \bullet \gamma'' = \gamma \bullet (\gamma' \bullet \gamma'')$ und $(r_x \bullet r_x) \bullet r_x \equiv r_x \bullet (r_x \bullet r_x)$

(b) $\gamma \bullet \gamma^{-1} = r_x$ und $r_x \bullet r_x^{-1} \equiv r_x$

(c) $(\gamma^{-1})^{-1} = \gamma$ und $(r_x^{-1})^{-1} \equiv r_x$

Beweis Mit (=I), Details kommen vielleicht noch.

Wie in der Topologie das Bild eines Weges unter einer stetigen Abbildung wieder ein Weg ist, kann auch hier zu jedem Pfad in einem Typ ein Pfad im Bild einer Funktion zugeordnet werden. Ganz untopologisch kann man die folgende Aussage dahingehend verstehen, dass Bilder gleicher Terme unter einer Funktion wieder gleiche Terme sind.

Definition + Bemerkung 1.5.6
Sei $f : A \to B$ eine beliebige Funktionen. Jedem $\gamma : x =_A y$ kann ein $f(\gamma) : f(x) = f(y)$ derart zugeordnet werden, dass $f(r_x) \equiv r_{f(x)}$ gilt.

Beweis Mit (=I): $f(\gamma) :\equiv i_=(a \mapsto r_{f(a)})$, Details kommen vielleicht noch.

Lemma 1.5.7 (Funktorialität)
Seien $f : A \to B$ und $x, y, z : A$. Für $\gamma : x = y$ und $\gamma' : y = z$ gilt:

$$f(\gamma \bullet \gamma') = f(\gamma) \bullet f(\gamma')$$

Beweis (=I) mit Reflexivitätsfall $a \mapsto (\gamma' : a = z) \mapsto r_{f(\gamma')}$.

Zum Abschluss möchten wir uns nun noch ein wenig mit verschiedenen Termen beschäftigen:

Definition 1.5.8
Seien $a, a' : A$ gegeben. Wir sagen, dass a und a' verschieden sind, wenn der Typ $a \neq_A a' :\equiv a =_A a' \to 0$ einen Term hat.

Beispiel 1.5.9
Die Terme $*_1$ und $*_2$ sind verschieden – der Typ $\neg(*_1 =_2 *_2)$ hat einen Term.
Um das einzusehen betrachten wir den abhängigen Typen $P : 2 \to 2 \to U$, der für Paare von Termen von 2 der Typ 0 ist, wenn sie ungleich sind und sonst 1 ist. P sei durch doppeltes Anwenden von (2R) gegeben.
Nun kann direkt durch (=I) gezeigt werden, dass es einen Term in

$$\prod_{x,y:2} x = y \to P(x, y)$$

gibt, welcher ausgewertet bei $(*_1, *_2)$ die Behauptung ist.

1.6 Natürliche Zahlen

Es gibt den Typ $\mathbb{N}$ der Natürlichen Zahlen mit den beiden Konstruktoren:

$$\text{Es gibt den Term } 0{:}\mathbb{N} \tag{$\mathbb{N}$K1}$$

$$\text{Für jeden Term } n{:}\mathbb{N} \text{ gibt es einen Term } S(n){:}\mathbb{N}. \tag{$\mathbb{N}$K2}$$

Beispiel 1.6.1
$1 :\equiv S(0)$, $2 :\equiv S(S(0)) \equiv S(1)$, $3 :\equiv S(2)$, …

Hier ist die Rekursionsregel tatsächlich rekursiv:

$$\begin{aligned}&\text{Es gibt } r_{\mathbb{N}}{:}C \to (C \to C) \to (\mathbb{N} \to C)\\ &\text{mit } r_{\mathbb{N}}(c_0, c_s)(0) \equiv c_0 \text{ und } r_{\mathbb{N}}(c_0, c_s)(S(n)) \equiv c_s(r_{\mathbb{N}}(c_0, c_s)(n)).\end{aligned} \tag{$\mathbb{N}$R}$$

Die Rekursionsregel liefert also für einen beliebigen Typ C eine Funktion $f{:}\mathbb{N} \to C$ sodass $f(0)$ der Vorgabe c_0 und $f(S(n))$ dem Bild der Rekursionsvorschrift c_s unter $f(n)$ entspricht.

Beispiel 1.6.2
$r_{\mathbb{N}}(0, (k{:}\mathbb{N}) \mapsto S(S(k)))$ ist eine Funktion des Typs $\mathbb{N} \to \mathbb{N}$, die ihr Argument verdoppelt.

Definition 1.6.3
Hier wird $C \equiv \mathbb{N} \to \mathbb{N}$ verwendet um die Addition über die Funktion „$k \mapsto k + \_$“ zu definieren. Mit der Rekursionregel ($\mathbb{N}$R) kann das wie folgt umgesetzt werden:

$$\_ + \_ :\equiv r_{\mathbb{N}}(1_{\mathbb{N}}, (f{:}\mathbb{N} \to \mathbb{N}) \mapsto k \mapsto S(f(k)))$$

Beispiel 1.6.4
Für beliebiges $n{:}\mathbb{N}$ gilt:

$$S(n) \equiv S(1_{\mathbb{N}}(n)) \equiv S(n+0) \equiv (k \mapsto S(k+0))(n) \equiv (k' \mapsto k' + (S(0)))(n) \equiv n + 1$$

Mit der unten vorgestellten Induktionsregel könnte man nun viele bekannte Eigenschaften der Addition auf den natürlichen Zahlen für die eben definierte Addition beweisen. Wir wollen beispielhaft die Assoziativität zeigen und damit gleichzeitig ein Anwendungsbeispiel der Induktionsregel für $\mathbb{N}$ geben.
Um eine Aussage $P{:}\mathbb{N} \to U$ zu zeigen, reicht es, die Aussage für $0{:}\mathbb{N}$ zu zeigen und für alle $n{:}\mathbb{N}$ aus einem Term in $P(n)$ einen Term in $P(S(n))$ zu konstruieren. Das ist genau das gleiche, wie den Induktionsanfang und Induktionsschritt für P zu zeigen. Ein Beweisterm für den Induktionschritt ist ein Term des Typs $\prod_{n:\mathbb{N}} P(n) \to P(S(n))$.
Insgesamt kann die Induktionsregel folgendermaßen formuliert werden:

$$\begin{aligned}&\text{Es gibt } i_{\mathbb{N}}{:}P(0) \to \left(\prod_{n:\mathbb{N}} P(n) \to P(S(n))\right) \to \prod_{n:\mathbb{N}} P(n)\\ &\text{mit } i_{\mathbb{N}}(p_0, p_s)(0) \equiv p_0 \text{ und } i_{\mathbb{N}}(p_0, p_s)(S(n)) \equiv p_s(i_{\mathbb{N}}(p_0, p_s)(n)).\end{aligned} \tag{$\mathbb{N}$I}$$

Nun, wie angekündigt, die Assoziativität von „+“:

Bemerkung 1.6.5
„+" ist eine assoziative Verknüpfung auf $\mathbb{N}$, das heißt es gilt:

$$\prod_{a,b,c:\mathbb{N}} (a+b)+c = a+(b+c)$$

Beweis Wir machen eine Induktion über a. Das heißt, dass das P aus der Induktionsregel ($\mathbb{N}$I) durch $a \mapsto \prod_{b,c:\mathbb{N}}(a+b)+c = a+(b+c) : \mathbb{N} \to U$ gegeben ist. Der Induktionsanfang ist also ein Term in $P(0) \equiv \prod_{b,c:\mathbb{N}}(0+b)+c = 0+(b+c)$. Solch ein Term ist durch $a \mapsto r_{b+c}$ gegeben, weil $(0+b)+c$ und $0+(b+c)$ urteilsmäßig gleich $b+c$ sind:

$$(0+b)+c \equiv (1_{\mathbb{N}}(b))+c \equiv b+c \equiv 1_{\mathbb{N}}(b+c) \equiv 0+(b+c)$$

Zu zeigen ist also noch, dass es einen Term in

$$\prod_{n:\mathbb{N}}(n+b)+c = n+(b+c) \to (S(n)+b)+c = S(n)+(b+c)$$

gibt. Dazu wenden wir zunächst auf eine Gleichheit $\gamma : (n+b)+c = n+(b+c)$ die Nachfolgerfunktion $S : \mathbb{N} \to \mathbb{N}$ an, um eine Gleichheit $S(\gamma) : S((n+b)+c) = S(n+(b+c))$ zu erhalten. Diese Gleichheit $S(\gamma)$ verbindet bis auf urteilsmäßige Gleicheiten, die per Definition der Addition gelten, bereits die richtigen Terme:

$$(S(n)+b)+c \equiv S(n+b)+c \equiv S((n+b)+c) = S(n+(b+c)) \equiv S(n)+(b+c)$$

Genauso wie $\mathbb{N}$ lässt sich auch $\mathbb{N}_{>0}$, der Typ der positiven natürlichen Zahlen konstruieren – mit dem einzigen Unterschied, dass es anstelle des Konstruktors $0 : \mathbb{N}$ einen Konstruktor $1 : \mathbb{N}_{>0}$ gibt und Rekursion und Induktion entsprechend bei 1 anfangen.

Bemerkung 1.6.6
Die Inklusion $\iota_{\mathbb{N}_{>0}} : \mathbb{N}_{>0} \to \mathbb{N}$ ist gegeben durch:

$$\iota_{\mathbb{N}_{>0}} :\equiv r_{\mathbb{N}_{>0}}(S(0), n \mapsto S(n))$$

Im weiteren Verlauf wollen wir $\mathbb{N}_{>0}$ lediglich zur Definition des Typs der ganzen Zahlen $\mathbb{Z}$ verwenden, was sich auch unter Verwendung von $\mathbb{N}$ bewerkstelligen ließe, allerdings geringfügig unschöner wäre.
Es gibt einen Typ $\mathbb{Z}$ mit drei Konstruktoren:

Für $n : \mathbb{N}_{>0}$ gibt es $+n : \mathbb{Z}$. ($\mathbb{Z}$K1)

Es gibt $0 : \mathbb{Z}$. ($\mathbb{Z}$K2)

Für $n : \mathbb{N}_{>0}$ gibt es $-n : \mathbb{Z}$. ($\mathbb{Z}$K3)

Die Rekursionfunktion hat folgenden Typ:

$$r_{\mathbb{Z}} : (\mathbb{N}_{>0} \to C) \to C \to (\mathbb{N}_{>0} \to C) \to (\mathbb{Z} \to C)$$

Beispiel 1.6.7
Für eine ganze Zahl k, also einen Term $k:\mathbb{Z}$ können wir Positiv- und Negativteil etwa wie folgt definieren:

$$\begin{aligned} k_+ &:\equiv r_{\mathbb{Z}}(\iota_{\mathbb{N}_{>0}}, 0, k \mapsto 0)(k):\mathbb{N} \\ k_- &:\equiv r_{\mathbb{Z}}(k \mapsto 0, 0, \iota_{\mathbb{N}_{>0}})(k):\mathbb{N} \end{aligned}$$

Ziel des Ende dieses Abschnitts ist es, die Vorgänger- und Nachfolgerfunktion $P, S:\mathbb{Z} \to \mathbb{Z}$ zu definieren, weil wir diese später noch brauchen werden.

Die Nachfolgerfunktion S ist auf den positiven Zahlen, das heißt Termen $+n:\mathbb{Z}$ leicht mittels der Nachfolgerfunktion S auf $\mathbb{N}_{>0}$ zu definieren. Erfreulicherweise ist die Definition der Nachfolgerfunktion auf den Negativen Zahlen – wie gleich zu sehen sein wird – kaum schwieriger und kommt damit insgesamt ohne die üblicherweise für Nachfolgerfunktionen auf $\mathbb{N}$ verwendeten Tricks aus.

$$S :\equiv r_{\mathbb{Z}}(n \mapsto +S(n), +1, r_{\mathbb{N}_{>0}}(0, k \mapsto -S(k_-)))$$

Analog kann die Vorgängerfunktion definiert werden:

$$P :\equiv r_{\mathbb{Z}}(r_{\mathbb{N}_{>0}}(0, k \mapsto +S(k_+)), -1, n \mapsto -S(n))$$

Nun wollen wir noch zeigen, dass S und P zueinander „invers“ sind. Dazu werden wir die für eine Aussage $Q:\mathbb{Z} \to U$ gegebene Induktionsfunktion für $\mathbb{Z}$ verwenden:

$$i_{\mathbb{Z}}: \left(\prod_{n:\mathbb{N}_{>0}} Q(+n)\right) \to Q(0) \to \left(\prod_{n:\mathbb{N}_{>0}} Q(-n)\right) \to \prod_{k:\mathbb{Z}} Q(k)$$

Lemma 1.6.8
Es gilt:

$$\prod_{k:\mathbb{Z}} (P \circ S)(k) = k \text{ und } \prod_{k:\mathbb{Z}} (S \circ P)(k) = k$$

Beweis Wir beweisen das per $\mathbb{Z}$-Induktion über k. Dazu brauchen wir zuerst einen Term in $\prod_{n:\mathbb{N}_{>0}} (P \circ S)(+n) = +n$. Verwenden von urteilsmäßigen Gleichheiten für S und P ergibt, dass die rechte Seite der Gleichung ebenfalls $+n$ ist – ein passender Term ist somit durch $n \mapsto r_{+n}$ gegeben.
Ein Term in $(P \circ S)(0) = 0$ ist durch die Reflexivität r_0 gegeben, wie sich ebenfalls durch Auflösen der Definitionen zeigt.
Den letzen Fall, $\prod_{n:\mathbb{N}_{>0}} (P \circ S)(-n) = -n$, klären wir mit $\mathbb{N}_{>0}$-Induktion über n. Interessant ist dabei der Induktionsschritt. Zu konstruieren ist eine Gleichheit in $(P \circ S)(-S(n)) = -S(n)$. Da links ein Nachfolger im Argument steht, kann die urteilsmäßige Gleichheit für die Nachfolgerfunktion auf den negativen Zahlen angewandt werden. Es bleibt also eine Gleichheit in $P(-n) = -S(n)$ zu finden. Eine solche ergibt sich aber aus der Definition von P.
Auf den sehr ähnlichen Beweis der zweiten Aussage wird hier verzichtet.

1.7 Abhängige Summen

Wie schon im Abschnitt zum abhängigen Produkt bemerkt wurde, kann dieses in manchen Situationen als allquantifizierte Aussage verstanden werden. Das Analogon für existenzquantifizierte Aussagen ist die abhängige Summe:

$$\text{Für } P : A \to U \text{ gibt es den Typ } \sum_{a:A} P(a). \tag{$\sum$Z}$$

Bemerkung 1.7.1
$\sum_{a:A} P(a)$ kann *nicht* wie in klassischer Logik mittels $\prod$ definiert werden. Dazu werden wir später vielleicht noch ein Beispiel sehen.

Ein kanonischer Term von $\sum_{a:A} P(a)$ ist ein Paar (a, p_a), bei dem die zweite Komponente Term eines Typs ist, der von der ersten Komponente abhängt:

$$\text{Für } a : A \text{ und } p_a : P(a) \text{ gibt es einen Term } (a, p_a) : \sum_{a:A} P(a). \tag{$\sum$K}$$

Interpretiert man $\sum$ als Existenzquantor, und ist ein Paar $(a, p_a) : \sum_{a:A} P(a)$ gegeben, so ist die Existenzaussage wahr, a ist das laut der Aussage existierende Objekt und p_a ein Nachweisterm dafür, dass die Aussage $P(a)$ gilt.
Für abhängige Summen gibt es die üblichen Rekursions- und Induktionsregeln. Wir wollen uns nun die Rekursionsregel anschauen:

$$\begin{aligned}&\text{Für einen Typ } C \text{ gibt es } r_{\sum} : \left(\prod_{a:A} P(a) \to C\right) \to \left(\left(\sum_{a:A} P(a)\right) \to C\right)\\ &\text{mit } r_{\sum}(f)(a, p_a) \equiv f(a, p_a).\end{aligned} \tag{$\sum$R}$$

Beispiel 1.7.2
Mit ($\sum$R) lässt sich die Projektion auf A definieren:

$$\pi_1 :\equiv r_{\sum}(a \mapsto x \mapsto a)$$

Jedoch reicht ($\sum$R) nicht aus, um die Projektion auf den zweiten Faktor zu definieren, weil diese Werte in $P(a)$ – und damit in einem abhängigen Typ – annehmen muss. Dazu können wir die $\sum$-Induktion verwenden:

$$\begin{aligned}&\text{Für einen Typ } C : \left(\sum_{a:A} P(a)\right) \to U \text{ gibt es}\\ &i_{\sum} : \left(\prod_{a:A} \prod_{p:P(a)} C((a,p))\right) \to \left(\prod_{x:\sum_{a:A} P(a)} C(x)\right)\\ &\text{mit } i_{\sum}(f)(a, p_a) \equiv f(a, p_a).\end{aligned} \tag{$\sum$I}$$

Beispiel 1.7.3
Nun können wir die zweite Projektion definieren:

$$\pi_2 :\equiv i_{\sum}(a \mapsto p \mapsto p) : \prod_{x:\sum_{a:A} P(a)} P(\pi_1(x))$$

1.8 Ordnung im soweit angehäuften Chaos

Dieser letzte Abschnitt des Kapitels zur Typentheorie soll das ein oder andere nochmal geordnet darstellen. Als Erstes können wir nun die Tabelle der Korrespondenz zwischen Aussagen und Typen vervollständigen:

	Korrespondenz zwischen Aussagen und Typen
Typ	Aussage
0	Falsch.
1	Wahr.
$\neg A$	Nicht A.
$A \to B$	Aus A folgt B.
$A \times B$	A und B.
$A + B$	A oder B.
$\sum_{a:A} P(a)$	Es gibt ein $a:A$, sodass $P(a)$ gilt.
$\prod_{a:A} P(a)$	Für alle $a:A$ gilt $P(a)$.

Alle Typen in der Tabelle – mit Ausnahme des abhängigen Produkts – haben wir als *Induktive Typen* definiert. Ein Induktiver Typ ist im Wesentlichen durch eine endliche Liste von Konstruktoren $K_1, \dots, K_n$ gegeben. Wir wollen hier die Konstruktoren $K_1, \dots, K_n$ mit ihren Typen identifizieren – auch wenn es in Echt mehrere Konstruktoren gleichen Typs geben kann, wie etwa $-$ und $+$ beim Typ $\mathbb{Z}$. Die Konstruktortypen $K_1, \dots, K_n$ dürfen nur durch wiederholtes Anwenden der Funktionstypzusammensetzung oder der abhängigen Variante entstehen, sodass schließlich jeder Konstruktor Werte im zu definierenden Typ annimmt und der zu definierende Typ nur *positiv auftritt*. Letzteres verbietet beispielsweise Konstruktoren der Form $(A \to 0) \to A$, wenn A der zu definierende Typ ist, weil das erste Auftreten von A hier nicht positiv ist. Genauer bedeutet diese Positivität, dass für einen Konstruktortyp $A_1 \to A_2 \to \dots \to A_n \to A$ der zu definierende Typ A in jedem der Typen A_i nur als Ziel vorkommen darf. Im allgemeinen abhängigen Fall ist der Konstruktortyp $\prod_{a_1:A_1} \dots \prod_{a_n:A_n(a_1,\dots,a_{n-1})} A(a_1, \dots, a_n)$ und die Positivität besagt hier, dass $A(\_)$ nur als Zieltyp in den Typen $A_1, \dots, A_n(\_)$ auftaucht. Bis jetzt ist der Typ $\_ =_A \_ : A \times A \to U$ der einzige abhängige Typ, den wir als Induktiven Typ definiert haben.

Alle Regeln

Alle bisher aufgeführten Regeln sind entweder

(a) Zusammensetzung, Konstruktion, Rekursion oder Induktion Induktiver Typen

(b) Die Regeln für das abhängige Produkt

(c) Die Regeln für Universen

2 Homotopietypentheorie

2.1 Äquivalenzen und Univalenz

Zunächst möchten wir uns ein wenig der namensgebenden Relation Homotopie widmen. Erstaunlicherweise wird es erst durch Erweiterung der soweit eingeführten Typentheorie möglich sein zu beweisen, dass die folgendermaßen definierte Homotopie äquivalent zur Gleichheit von Funktionen ist:

Definition 2.1.1
Zwei Funktionen $f, g : A \to B$ heißen *homotop*, wenn es einen Term in

$$f \sim g :\equiv \prod_{a:A} f(a) = g(a)$$

gibt. Ein Term von $f \sim g$ heißt *Homotopie* von f nach g.

Bemerkung 2.1.2
Homotopie ist reflexiv, symmetrisch und transitiv.

Beweis 2.3.1 wird uns erlauben, das aus den entsprechenden Eigenschaften der Gleichheit zu folgern.

Nun können wir uns dem zentralen Begriff der Äquivalenz zuwenden:

Definition 2.1.3
Sei $f : A \to B$.

(a) Eine Abbildung $g : B \to A$ heißt *Quasiinverse* von f, wenn $g \circ f \sim 1_A$ und $f \circ g \sim 1_B$ gilt.

(b) Die Abbildung f heißt *Äquivalenz*, wenn es eine Quasiinverse zu f gibt.

(c) Wenn es eine Äquivalenz $e : A \to B$ gibt, dann sagt man, A ist *äquivalent* zu B.

Beispiel 2.1.4

(a) Die Identität 1_A ist stets eine Äquivalenz, denn: $1_A \circ 1_A \equiv 1_A \sim 1_A$.

(b) Die „Vertauschungsabbildung“ $s : 2 \to 2$ ist eine Äquivalenz, die – wie die Identität – ihre eigene Quasiinverse ist. Um das einzusehen kann man die Bilder $(s \circ s)(*_1) \equiv *_1$ und $(s \circ s)(*_2) \equiv *_2$ berechnen. Das allein reicht allerdings noch nicht aus – es ist noch 2-Induktion auf das soeben Errechnete anzuwenden um zur Aussage $\prod_{x:2}(s \circ s)(x) = x$ zu gelangen.

(c) Der sogenannte *Transport* aus 1.5.2(b). Zu $\mathcal{T}_\gamma$ gibt es die Quasiinverse $\mathcal{T}_{\gamma^{-1}}$. Ein möglicher einfacher Beweis dieser Tatsache ist eine Induktion über γ.

(d) Zu jedem Typ A kann der Typ aller Gleichheiten in A gebildet werden: $\sum_{x,y:A} x =_A y$. Eng verwandt mit der Regel (=I) ist die Erkenntnis, dass die Abbildung $a \mapsto (a,(a,r_a)) : A \to \sum_{x,y:A} x =_A y$ eine Äquivalenz ist.
Als Quasiinverse kann die Abbildung verwendet werden, die eine Gleichheit auf ihren Anfangspunkt abbildet: $(x,(y,\gamma)) \mapsto x$. Im Wesentlichen fehlt dann nur noch eine Gleichheit $(y,\gamma) = (x,r_x)$ für beliebiges y und Gleichheiten $\gamma : x = y$. Letzteres kann direkt mit (=I) konstruiert werden.

(e) Die Vorgänger- und Nachfolgerfunktion aus 1.6 sind nach 1.6.8 quasiinvers und damit Äquivalenzen des Typs $\mathbb{Z} \to \mathbb{Z}$.

Aufgaben/Bemerkungen 2.1.5

(a) Äquivalenz von Typen ist eine reflexive, symmetrische und transitive Relation.

(b) Wenn für eine Äquivalenz $f : A \to B$ eine Abbildung g mit $f \circ g \sim 1_B$ existiert, dann gilt auch $g \circ f \sim 1_A$. Die duale Aussage gilt ebenfalls. Weiter ist damit ein Grund gegeben, bei der Diskussion von Äquivalenz nicht mit Links- und Rechtsquasiinversen, sondern lediglich mit Quasiinversen zu arbeiten.

Gelegentlich ist es nützlich, für bestimmte Sorten von Aussagen eine feste Konstruktion anzugeben. Wir werden später noch sehen, dass es zu logisch äquivalenten Aussagen mehrere nicht äquivalente Typen gibt, die diese Aussagen formalisieren. Vorzugsweise verwendet man für Aussagen eine spezielle Sorte von Typen, die man sich an dieser Stelle am Besten als leer oder kontrahierbar[1] vorstellt.
Der Typ in der folgenden Definition ist von eben erwähnter Sorte – was nicht der Fall wäre, wenn direkt die Aussage „hat eine Quasiinverse“ verwendet worden wäre.

Definition 2.1.6

(a) Für $f : A \to B$ sei

$$\mathcal{E}_f :\equiv \sum_{g,h:B\to A} (f \circ g \sim 1_B) \times (h \circ f \sim 1_A)$$

(b) Nun lässt sich für Typen A und B durch

$$A \simeq B :\equiv \sum_{f:A\to B} \mathcal{E}_f$$

der Typ der Äquivalenzen von A nach B definieren. Andererseits entspricht dieser Typ auch der Aussage „A ist äquivalent zu B“.

[1] Was das genau bedeutet wird Thema des nächsten Abschnitts sein. Es ist durchaus gerechtfertigt, sich darunter Kontrahierbarkeit im topologischen Sinn vorzustellen.

Um die Lesbarkeit zu erhöhen, unterdrücken wir gelegentlich die Zusatzdaten eines Terms $(f,(h_1,h_2)){:}A \simeq B$ und schreiben lediglich $f{:}A \simeq B$.

Aufgabe 2.1.7

(a) In klassischer Topologie gibt es ebenfalls den Begriff der Homotopie. Man mache sich für den Kreis $S^1 \subseteq \mathbb{C}$ klar, dass $k \mapsto (z \mapsto z^k){:}\mathbb{Z} \to (S^1 \simeq S^1)$ eine Äquivalenz ist[2]. „Klarmachen" ist hier so zu verstehen, dass man sich verdeutlicht, was die Aussage bedeutet und nicht etwa versucht einen formalen Beweis anzugeben.

(b) Weiter mache man sich klar, dass $1_{S^1} \simeq 1_{S^1}$ ebenfalls äquivalent zu $\mathbb{Z}$ ist.

(c) Schließlich mache man sich klar, dass der Typ $\mathcal{E}_f$ für jede Äquivalenz $f{:}1_{S^1} \simeq 1_{S^1}$ kontrahierbar ist, der Typ $\sum_{g:S^1\to S^1}(g \circ f \sim 1_{S^1}) \times (f \circ g \sim 1_{S^1})$ jedoch nicht!

Bemerkung 2.1.8
Gleiche Typen sind äquivalent.

Beweis Man betrachte den abhängigen Typen $1_U{:}U \to U$. Sind nun A und B gleiche Typen, dann gibt es $\gamma{:}A = B$. Aus 2.1.4(c) wissen wir, dass $\mathcal{T}_\gamma{:}A \to B$ eine Äquivalenz ist.

Eine Konsequenz der letzten Bemerkung ist, dass es für beliebige Typen $A, B{:}U$ stets eine Funktion $\gamma \mapsto \mathcal{T}_\gamma{:}A = B \to A \simeq B$ gibt. Erstaunlicherweise steht die soweit vorgestellte Theorie nicht mit der Annahme in Konflikt, dass diese Funktion eine Äquivalenz ist.[3] Diese Annahme wird *Univalenzaxiom* genannt und im Folgenden präzisiert um schließlich als Axiom in unsere Typentheorie aufgenommen zu werden.

Definition 2.1.9
Ein Universum U heißt *univalent*, wenn die Funktion $\gamma \mapsto \mathcal{T}_\gamma{:}A = B \to A \simeq B$ eine Quasiinverse $u{:}A \simeq B \to A = B$ hat.

Axiom
Die Universen $U_1, U_2, \ldots$ sind univalent.

Wir können nun auf Äquivalenzen stets die Funktion u anwenden, um eine Gleichheit zwischen den beiden Typen zu erhalten. Durch die Wahl von u als Quasiinverse der Funktion $\gamma \mapsto \mathcal{T}_\gamma$ ergeben sich nützliche Eigenschaften:

Bemerkung 2.1.10
Seien Äquivalenzen $f{:}A \to B$ und $g{:}B \to C$ gegeben. Es gilt:

(a) $u(1_A) = r_A$

(b) $u(g \circ f) = u(f) \bullet u(g)$

---

[2] Das ist leider nicht einfach wird aber andererseits in vielen einführenden Texten zur Topologie thematisiert.

[3] Das wurde von Vladimir Voevodsky und Anderen durch Konstruktion eines Models in Mengenlehre-basierter Mathematik gezeigt. Details findet man in [LKV14].

(c) $u(f)^{-1} = u(f^{-1})$, wobei f^{-1} eine Funktion mit $f^{-1} \circ f = 1_A$ sei.

Beweis (a) In 1.5.2(b) wurde festgelegt, dass der Transport für $\gamma \equiv r_A$ die urteilsmäßig gleich der Identität ist. Daher können wir $u(1_A)$ als $u(\mathcal{T}_{r_A})$ schreiben. Nun ist $u(\mathcal{T}_{r_A}) \equiv u((\gamma \mapsto \mathcal{T}_\gamma)(r_A)) \equiv u \circ (\gamma \mapsto \mathcal{T}_\gamma)(r_A) = r_A$, wobei die letzte Gleichheit dadurch gegeben ist, dass u und $\gamma \mapsto \mathcal{T}_\gamma$ quasiinvers sind.

(b) Durch (=I) zeigt man $\mathcal{T}_{\gamma \bullet \gamma'} = \mathcal{T}_{\gamma'} \circ \mathcal{T}_{\gamma'}$. Dann gilt wieder unter reichlich Ausnutzung der Quasiinversität: $u(g \circ f) = u(\mathcal{T}_{u(g)} \circ \mathcal{T}_{u(f)}) = u(\mathcal{T}_{u(f) \bullet u(g)}) = u(f) \bullet u(g)$

(c) Mit (b) ergibt sich: $r_A = u(1_A) = u(f^{-1} \circ f) = u(f) \bullet u(f^{-1})$. Durch Linksanhängung von $u(f)^{-1}$ ergibt sich die Behauptung.

Verblüffenderweise hat nun also etwa der Typ $2 = 2$ zwei verschiedene Elemente – nämlich r_2 und $u(s)$, wenn s die Äquivalenz ist, die die beiden Terme von 2 vertauscht, denn wenn $\mathcal{T}_{r_2} \equiv 1_2 = s$ gälte, dann wäre beispielsweise auch $1_2(*_1) = s(*_1)$, also $*_1 = *_2$ – was bereits in 1.5.9 widerlegt wurde.

2.2 Kontrahierbarkeit

Definition 2.2.1
Für einen Typ A sei $\mathcal{C}(A) :\equiv \sum_{a_0 : A} \prod_{a:A} a_0 = a$. Wenn $\mathcal{C}(A)$ gilt, dann heißt A *kontrahierbar*. Terme wie a_0 werden auch *Kontraktionszentrum* und Terme in $\prod_{a:A} a_0 = a$ *Kontraktionen* genannt.

Interpretiert man Gleichheiten von Termen als Wege zwischen Punkten, so scheint obige Definition auf den ersten Blick eher Wegzusammenhang statt Kontrahierbarkeit zu beschreiben. Dass es sich trotzdem um Kontrahierbarkeit handelt, kommt durch den Unterschied zwischen $\prod$ und dem Allquantor zustande – hier wird eine stetige Familie von Wegen gefordert, die zu einer Kontraktion zusammengesetzt werden können.

Beispiel 2.2.2
(a) Der Typ 1 ist kontrahierbar. Einen Term von $\prod_{x:1} x = *$ haben wir bereits in 1.5.1(a) konstruiert. Nennen wir diesen p so gilt: $(*, p) : \mathcal{C}(1)$.

(b) Der Typ 2 ist wegen 1.5.9 nicht kontrahierbar.

Bemerkung 2.2.3
Ein Typ A ist genau dann kontrahierbar, wenn A äquivalent zu 1 ist.

Beweis Sei ohne Einschränkung $(a_0, k) : \mathcal{C}(A)$, dann sind $a \mapsto * : A \to 1$ und $x \mapsto a_0 : 1 \to A$ quasiinvers – die nötigen Homotopien sind durch k gegeben.
Sind andererseits $f : 1 \to A$ und $g : A \to 1$ quasiinvers, dann verwenden wir $a_0 :\equiv f(*)$ als Kontraktionszentrum. Weil f und g quasiinvers sind, gibt es eine Homotopie von $f \circ g$ nach 1_A, also $\tilde{k} : \prod_{a:A} f(g(a)) = a$. Wegen 2.2.2(a) gibt es $k_1 : \prod_{x:1} * = x$ und es gilt: $a \mapsto f(k_1(g(a))) \bullet \tilde{k}(a) : \prod_{a:A} a_0 = a$.

Nachdem wir nun in der letzten Bemerkung gesehen haben, wie wir Kontrahierbarkeit mittles Äquivalenz ausdrücken können, werden wir als nächsten Äquivalenzen mittels Kontrahierbarkeit der Fasern einer Abbildung ausdrücken:

Definition 2.2.4
Sei $f:A \to B$ eine Abbildung und $b:B$, dann ist durch

$$f_b :\equiv \sum_{a:A} f(a) = b$$

die *Faser* von f über b gegeben.

Satz 2.2.5
Eine Abbildung $f:A \to B$ ist genau dann eine Äquivalenz, wenn alle Fasern von f kontrahierbar sind.

Beweis Zunächst seien alle Fasern von $f:A \to B$ kontrahierbar. Es gibt also $k:\prod_{b:B} \mathcal{C}(f_b)$. Wir fahren fort, indem wir eine Quasiinverse $g:B \to A$ zu f konstruieren. Dazu sei $g(b)$ das durch k gegebene Kontraktionszentrum der Faser über b, also $g :\equiv b \mapsto \pi_1(k(b))$. Eine Homotopie $g \circ f \sim 1_A$ ist also direkt durch die Kontraktionen der Fasern gegeben. Die zweite Homotopie $f \circ g \sim 1_B$ kommt punktweise dadurch zustanden, dass $g(b)$ das Kontraktionszentrum der Faser f_b ist, also insbesondere selbst in dieser liegt und somit ein Pfad in $f(g(b)) = b$ existiert. Insgesamt ist g in diesem Fall eine Quasiinverse und damit f eine Äquivalenz.
Sei nun $f:A \to B$ eine Äquivalenz mit Quasiinverser $g:B \to A$. Sei weiter $b:B$. Wir möchten zeigen, dass die Faser f_b kontrahierbar ist. Als Kontraktionszentrum wählen wir $g(b)$. Dazu müssen wir uns noch überzeugen, dass $g(b)$ in f_b liegt: Wir haben eine Gleichheit $f(g(b)) = b$ durch die Quasiinversität von g gegeben – das reicht. Es bleibt noch eine Kontraktion aller $a:A$ mit $\gamma : f(a) = b$ auf $g(b)$ zu kontruieren. Dazu wird γ in die Faser gehoben: $g(\gamma):g(f(a)) = g(b)$. Und schließlich die zweite Homotopie $h:1_A \sim g \circ f$ bei a ausgewertet und an $g(\gamma)$ angehängt: $h(a) \bullet g(\gamma):a = g(b)$.

Definition 2.2.6

(a) Kontrahierbare Typen heißen auch (-2)-Typen. Weiter ist ein (n+1)-Typ ein Typ A, sodass alle $x =_A y$ n-Typen sind.

(b) Speziell heißen (-1)-Typen *bloße Aussagen* und 0-Typen *Mengen*.

Bemerkung 2.2.7

(a) Unter bloßen Aussagen darf man sich Typen vorstellen die auf den Wahrheitswert der Aussage die sie darstellen reduziert sind. Unter Mengen Mengen.

(b) Jeder kontrahierbare Typ ist eine bloße Aussage.

Beweis (b) Seien $x, y:A$ und ohne Einschränkung $(a_0, k):\mathcal{C}(A)$. Es ist ein Term des Typs $\prod_{x,y:A} \mathcal{C}(x = y)$ zu konstruieren. Als Kontraktionszentrum von $x = y$ verwenden wir $\gamma_0 :\equiv k(x) \bullet k(y)^{-1}$. Nun ist eine Kontraktion, also ein Term in

$k_{\gamma_0} \prod_{\gamma : x=y} \gamma = \gamma_0$ zu konstruieren. Mittels (=I) reicht es für $\gamma \equiv r_x$ einen Term in $\gamma \equiv r_x = \gamma_0 \equiv k(x) \bullet k(x)^{-1} = r_x$ anzugeben – was durch die Wahl r_{r_x} erledigt ist.

Beispiel 2.2.8

(a) Der Typ 1 ist kontrahierbar und damit auch eine bloße Aussage.

(b) Der Typ 0 ist nicht kontrahierbar, aber trotzdem eine bloße Aussage.

(c) Der Typ $\mathcal{E}_f$, der für eine Abbildung $f : A \to B$ die Aussage „f ist eine Äquivalenz" realisiert, ist stets eine bloße Aussage. Auf einen Beweis verzichten wir hier leider.

Für beliebige Typen A gibt es einen Typ $\|A\|$, der eine bloße Aussage ist, und aus A stets $\|A\|$ und aus $\neg A$ stets $\neg\|A\|$ gefolgert werden kann. Näheres dazu findet man in [Uni13, Abschnitt 3.7]. Weiter können das Axiom vom Ausgeschlossenen Dritten und das Auswahlaxiom für bloße Aussagen problemlos bei Bedarf angenommen werden. Details hierzu gibt es in [Uni13, Abschnitt 3.4].

2.3 Höhere Induktive Typen

Wir haben in Kapitel 1 viele Induktive Typen gesehen: 0, 1, 2, $\mathbb{N}$, $\mathbb{Z}$, $x =_A y$, ...
Bei Höheren Induktiven Typen können Konstruktoren auch Werte in Gleichheitstypen annehmen. Wir werden wieder darauf verzichten eine allgemeine Konstruktionsvorschrift anzuschauen und stattdessen Beispiele betrachten[4].
Das *Intervall* I ist ein Höherer Induktiver Typ mit folgenden Konstruktoren:

$$0 : I \tag{IK0}$$

$$1 : I \tag{IK1}$$

$$s : 0 =_I 1 \tag{IKS}$$

Wie bei gewöhnlichen Induktiven Typen gibt es auch hier Rekursions- und Induktionsregeln. Neu ist bei der Rekursion, dass auch für die per Konstruktor gegebenen Gleichheiten eine Gleichheit im Zieltyp angegeben werden muss. Für einen beliebigen Typ A gilt also die Regel:

$$\begin{aligned} &\text{Für } a_0 : A,\ a_1 : A \text{ und } a_s : a_0 = a_1 \text{ gibt es eine Abbildung } f : I \to A \\ &\text{mit } f(0) \equiv a_0,\ f(1) \equiv a_1 \text{ und } f(s) = a_s \end{aligned} \tag{IR}$$

Das folgende Lemma erlaubt uns, aus punktweiser Gleichheit zweier Funktionen auf Gleichheit der Funktionen zu schließen:

Lemma 2.3.1
Sei $P : A \to U$ ein abhängiger Typ. Für $f, g : \prod_{a:A} P(a)$ folgt aus $\prod_{a:A} f(a) = g(a)$ bereits $f = g$.

[4] Die ohnehin nach Wissen des Autors nicht existiert; bzw. Gegenstand aktueller Forschung ist. Näheres: [Uni13, Chapter 6]

Beweis Seien $f, g : \prod_{a:A} P(a)$ und $p : \prod_{a:A} f(a) = g(a)$. Für $a : A$ lässt sich eine Abbildung $h_a : I \to P(a)$ durch $h_a(0) :\equiv f(a)$, $h_a(1) :\equiv g(a)$ und $h_a(s) :\equiv p(a)$ via (IR) definieren. Damit ist auch eine Abbildung $h : I \to \prod_{a:A} P(a)$ durch $h(i) :\equiv a \mapsto h_a(i)$ gegeben. Also gilt $h(s) : f = g$.

Das wollen wir sofort für eine überraschende Charakterisierung bloßer Aussagen verwenden:

Korollar 2.3.2 (zu 2.3.1)
Aus $\prod_{x,y:A} x = y$ folgt bereits, dass A eine bloße Aussage ist.

Beweis Zu zeigen ist die Kontrahierbarkeit der $x =_A y$. Sei $\mathfrak{p} : \prod_{x,y:A} x = y$ gegeben. Wir wählen als Kontraktionszentrum von $x =_A y$ die Gleichheit $\mathfrak{p}(x, y) \equiv: \gamma_0$. Es ist nun noch für ein beliebiges $\gamma : x =_A y$ ein $H : \gamma = \gamma_0$ zu konstruieren. Bildlich kann man sich H als kontrahierbare Fläche zwischen γ und γ_0 vorstellen, die wir im folgenden aus Wegen, die quer zu γ und γ_0 verlaufen, mit Hilfe von 2.3.1 zusammenkleben werden.
Für den letzten Schritt seien für die Gleichheiten γ und γ_0 Abbildungen $\tilde{\gamma}$ und $\tilde{\gamma_0}$ mit $\tilde{\gamma}(s) = \gamma$ und $\tilde{\gamma_0}(s) = \gamma_0$ gegeben. Es ist $\tilde{H} : \prod_{x:I} \tilde{\gamma}(x) = \tilde{\gamma_0}(x)$ gegeben durch $x \mapsto \mathfrak{p}(\widetilde{\gamma(x)}, \tilde{\gamma_0})$. Mit 2.3.1 erhalten wir ein $H : \gamma = \gamma_0$.

Aufgabe 2.3.3

(a) Zeige: I ist kontrahierbar.

(b) Zeige: $(I \to A) \simeq (1 \to A) \simeq A$ für beliebige Typen A. Folgere mit 2.1.4(d) die Äquivalenz $(I \to A) \simeq \sum_{x,y:A} x = y$.

Induktionsregeln für Höhere Induktive Typen sind leider etwas umständlicher als die bereits bekannten. Wollen wir für $P : I \to U$ die Aussage $\prod_{x:I} P(x)$ zeigen, so müssen wir – wie gehabt – zunächst Terme $p_0 : P(0)$ und $p_1 : P(1)$ angeben. Erwartungsgemäß muss das *Segment* $s : 0 =_I 1$ auf eine Gleichheit abgebildet werden, die p_0 und p_1 verbindet. Das lässt sich allerdings nicht direkt formulieren, weil p_0 und p_1 von unterschiedlichem Typ sind. Weiter sollte die Gleichheit zwischen p_0 und p_1 „über s liegen". Man könnte natürlich zu $\sum_{x:I} P(x)$ übergehen, um zumindest nach einer Gleichheit zwischen $(0, p_0)$ und $(1, p_1)$ fragen zu können – allerdings gibt es eine Definition, mit der sich leichter arbeiten lässt. Dabei wird die Existenz eines Terms $p_s : \mathcal{T}_s(p_0) = p_1$ gefordert. Die sollte teilweise klären, warum sich auf diese Art ausdrücken lässt, dass p_s eine Gleichheit „über s" ist. Weiter wird in Teil (d) eine Verallgemeinerung des Anwendens von Funktionen auf Gleichheiten vorgestellt, die es möglich machen wird, in der Induktionsregel zu fordern, dass p_s durch die induzierte Funktion auch wirklich realisiert wird.

Bemerkung 2.3.4
Sei $P : A \to U$.

(a) Für $a : A$, $p : P(a)$ und $\gamma : a = a'$ gibt es $\tilde{\gamma} : (a, p) = (a', \mathcal{T}_\gamma(p))$ in $\sum_{x:A} P(x)$.

(b) $\tilde{\gamma}$ „liegt über γ" im Sinne des Textes oben, d.h. $\pi_1(\tilde{\gamma}) = \gamma$.

(c) Liegt $\tilde{\gamma}$ über γ und $\tilde{\gamma'}$ über γ', dann liegt auch $\tilde{\gamma} \bullet \tilde{\gamma'}$ über $\gamma \bullet \gamma'$ (sofern γ und γ' überhaupt aneinandergehängt werden können).

(d) Für $x, y : A$, $p : \prod_{a:A} P(a)$ und $\gamma : x = y$ gibt es $p(\gamma) : \mathcal{T}_\gamma(x) =_{P(y)} y$.

Teil (a) sagt, dass es möglich ist, Gleichheiten $\gamma : x =_A y$ nach $\sum_{a:A} P(a)$ „hochzuheben“. Die Gleichheit $p_s : \mathcal{T}_s(p_0) = p_1$ ist dann sozusagen genau das noch fehlende Stück, da damit dann durch Teil (c) insgesamt eine Gleichheit über s gegeben ist. Außerdem liegt $\tilde{s} \bullet (1, p_s)$ immer noch über s, weil $(1, p_s)$ komplett in der Faser über $1 : I$ liegt.

Beweis (von 2.3.4) (a) Kann direkt mit (=I) gezeigt werden.

(b) Ebenfalls mit (=I) – r_a liegt über $\tilde{r_a} = r_{(a,p)}$, denn nach 1.5.6 gilt $\pi_1(r_{(a,p)}) \equiv r_a$.

(c) Folgt aus 1.5.7.

(d) (=I).

Die Induktionsregel lautet nun insgesamt:

$$\begin{aligned}&\text{Für } p_0 : P(0),\ p_1 : P(1) \text{ und } p_s : \mathcal{T}_s(p_0) = p_1 \text{ gibt es } p : \prod_{x:I} P(x)\\ &\text{mit } p(0) \equiv p_0,\ p(1) \equiv p_1 \text{ und } p_s = p(s).\end{aligned} \tag{II}$$

Anwendungen werden wir nie direkt sehen – bei der sehr ähnlichen Induktionsregel für den Kreis allerdings schon. Diesem werden wir uns nun zuwenden. Der *Kreis* ist ein Typ, der mit S^1 bezeichnet wird und folgende Konstruktoren hat:

$$* : S^1 \tag{S^1K}$$

$$l : * =_{S^1} * \tag{S^1KL}$$

Die Rekursionregel ist analog zu der des Intervalls:

$$\begin{aligned}&\text{Für } a_* : A \text{ und } a_l : a_* =_A a_* \text{ gibt es eine Abbildung } f : S^1 \to A\\ &\text{mit } f(*) \equiv a_* \text{ und } f(l) = a_l\end{aligned} \tag{S^1R}$$

Ebenso die Induktion für $P : S^1 \to U$:

$$\begin{aligned}&\text{Für } p_* : P(*) \text{ und } p_l : \mathcal{T}_l(p_*) = p_* \text{ gibt es } p : \prod_{x:I} P(x)\\ &\text{mit } p(*) \equiv p_* \text{ und } p_l = p(l).\end{aligned} \tag{S^1I}$$

Es gibt also eine Rekursionsabbildung $r_{S^1} : (a : A) \to a = a \to (S^1 \to A)$ und eine Induktionsabbildung $i_{S^1} : (p : P(*)) \to \mathcal{T}_l(p) = p \to \left(\prod_{x:S^1} P(x)\right)$.

Lemma 2.3.5
Es gilt:

$$(S^1 \to A) \simeq \sum_{a:A} a = a$$

Beweis Wir zeigen mittels 2.2.5, dass

$$e :\equiv f \mapsto (f(*), f(l)) : (S^1 \to A) \to \sum_{a:A} a = a$$

eine Äquivalenz ist. Wir müssen also für jede Faser e_x von e ein Kontraktionszentrum und eine Kontraktion angeben. Als Kontraktionzentrum über $x : \sum_{a:A} a = a$ wählen wir die durch S^1R gegebene Abbildung $r_{S^1}(\pi_1(x), \pi_2(x))$. Die Kontraktion selbst kann auch als Eindeutigkeit der S^1-Rekursion verstanden werden – jede Abbildung f mit $a_* = f(*)$ und $a_l = f(l)$ ist bereits gleich der durch S^1-Rekursion mit Eingangsdaten a_* und a_l gegebenen.
Es seien nun also $f : S^1 \to A$ in der Faser über (a_*, a_l) also $\gamma : a_* = f(*)$ und $H : a_l = \gamma \bullet f(l) \bullet \gamma^{-1}$. Weiter sei $g :\equiv r_{S^1}(a_*, a_l)$. Es ist $f = g$ zu zeigen. Wie so oft hilft auch hier (=I). Dazu lassen wir a_* und γ variieren. Dann ist, um (=I) anwenden zu können, nur noch der Fall $a_* \equiv f(*)$ zu zeigen. Sei $\tilde{H} : g(l) = f(l)$. Wir verwenden (S^1I) um daraus $H : \prod_{x:S^1} f(x) = g(x)$ zu konstruieren. Wir wissen bereits $r_{f(*)} : f(*) = g(*) \equiv f(*)$ – um (S^1I) anwenden zu können brauchen wir noch etwas in $\mathcal{T}_l(r_{f(*)}) = r_{f(*)}$. Dazu muss zuerst der Transport berechnet werden: $\mathcal{T}_l(r_{f(*)}) = f(l)^{-1} \bullet r_{f(*)} \bullet g(l)$ – was man für beliebige Gleichheiten mit (=I) zeigt. Mit $\tilde{H}$ etwa $g(l)$ ersetzen und vereinfachen liefert die Behauptung.

2.4 Der Schleifenraum von S^1

Ziel dieses Abschnitts ist es, zu beweisen, dass der Schleifenraum von S^1 äquivalent zu den ganzen Zahlen, also dem Typ $\mathbb{Z}$ ist. Ein Korollar dieser Aussage ist, dass die Fundamentalgruppe des Kreises isomorph zu $\mathbb{Z}$ ist – was gerne in Grundlagenvorlesungen zu mengentheoretischer Topologie gezeigt wird. Unterwegs werden einige klassische, recht anschauliche Begriffe auftauchen: die Windungszahl und die Universelle Überlagerung des Kreises. Beide lassen sich erstaunlich einfach definieren:

Definition 2.4.1

(a) Die *Universelle Überlagerung* $V : S^1 \to U$ von S^1 ist durch (S^1R) gegeben, wobei für $V(*)$ der Typ $\mathbb{Z} : U$ und für $V(l)$ die durch die Nachfolgerfunktion $S : \mathbb{Z} \to \mathbb{Z}$ gegebene Gleichheit $u(S) : \mathbb{Z} =_U \mathbb{Z}$ gewählt wird.

(b) Der *Schleifenraum* von S^1 ist gegeben durch $\Omega S^1 :\equiv * =_{S^1} *$.

(c) Die Windungszahl $\omega : \Omega S^1 \to \mathbb{Z}$ kann – mithilfe des Transports von V – direkt angegeben werden:

$$\omega :\equiv \gamma \mapsto \mathcal{T}_\gamma(0)$$

Es wird sich heraustellen, dass ω eine Äquivalenz ist. Eine Quasiinverse ist die Abbildung, die eine ganze Zahl $k : \mathbb{Z}$ auf l^k und damit den Weg, der k-mal S^1 durchläuft, abbildet. Wenn k negativ ist, werden Potenzen von l^{-1} verwendet. Wir wollen diese Abbildung mit l^- bezeichnen.

Wir wollen nun den Satz formulieren, der – wie eingangs schon erwähnt wurde – Gegenstand dieses Abschnitts ist:

Satz 2.4.2
Die Abbildungen

$$\omega{:}\Omega S^1 \to \mathbb{Z} \text{ und } l^-{:}\mathbb{Z} \to \Omega S^1$$

sind zueinander quasiinverse Äquivalenzen.

Wie schon so oft, wird es auch hier helfen die Aussage genau soweit zu verallgemeinern, dass (=I) angewandt werden kann. Die davon betroffenen Gleichheiten sind die Terme in ΩS^1. Um (=I) auf diese Gleichheiten anwenden zu können, muss die Aussage des Satzes zumindest auf $\prod_{x:S^1} * = x$ ausgeweitet werden. Dafür müssen zunächst die Abbildungen ω und l^- entsprechend fortgesetzt werden. Dafür können wir (S^1I) verwenden. Über $*$ verwenden wir ω und l^-. Um nun zu Abbildungen über ganz S^1 zu kommen, also Termen von $\prod_{x:S^1} * = x \to V(x)$ und $\prod_{x:S^1} V(x) \to * = x$, müssen wir noch die entsprechenden Transporte ausrechnen.
Genauer brauchen wir, um ω von $*$ auf ganz S^1 zu einer Abbildung $W{:}\prod_{x:S^1} * = x \to V(x)$ mittels (=I) fortzusetzen, eine Gleichheit in $\mathcal{T}_l(\omega) =_{\mathbb{Z}=\mathbb{Z}} \omega$. Es ist möglich für Transporte Rechenregeln für gängige Typen herzuleiten, worauf in dieser Vortragsreihe verzichtet wurde. Allerdings kann man an dieser Stelle recht gut eine Vermutung aufstellen, wie sich der Ausdruck $\mathcal{T}_l(\omega)$ umformen lässt. Hilfreich ist auch hier, die feste Gleichheit l durch eine beliebige $\gamma{:}* =_{S^1} x$ zu ersetzen. Dann ist

$$\mathcal{T}_\gamma{:}(* = * \to \mathbb{Z}) \to (* = x \to V(x)).$$

An dieser Stelle kann man raten, weil es nur eben diese eine naheliegende Möglichkeiten gibt, dass $\mathcal{T}_\gamma(f) \equiv \zeta \mapsto \mathcal{T}^V_\gamma f(\zeta \bullet \gamma^{-1})$ ist, was sich mit (=I) bestätigen lässt.
Damit ist das ursprüngliche Problem darauf reduziert, eine Gleichheit in

$$\zeta \mapsto \mathcal{T}^V_l \omega(\zeta \bullet l^{-1}) =_{\mathbb{Z}=\mathbb{Z}} \omega$$

zu finden. Der Ausdruck $\omega(\_ \bullet l^{-1})$ kann durch einen Blick auf die Definition von ω und der Eigenschaft „$\mathcal{T}_{\gamma\bullet\gamma'} = \mathcal{T}_{\gamma'} \circ \mathcal{T}_\gamma$“ von Transporten zu $\mathcal{T}^V_{l^{-1}}(\omega(\zeta))$ umgeformt werden. Die Transporte $\mathcal{T}^V_l$ und $\mathcal{T}^V_{l^{-1}}$ sind quasiinvers – womit das Problem gelöst ist. Damit können wir die Abbildung $W{:}\prod_{x:S^1} * = x \to V(x)$ mittels (S^1I) definieren, sodass $W(*) \equiv \omega$ gilt.
Nun wollen wir noch $L{:}\prod_{x:S^1} V(x) \to * = x$ konstruieren. Über $*$ können wir analog l^- verwenden und müssen damit noch eine Gleichheit der Form $\mathcal{T}_l(l^-) = l^-$ finden. Ähnlich wie oben kann wieder umgeformt werden:

$$\mathcal{T}_l(l^-) = \mathcal{T}^{*=x}_l(l^- \circ \mathcal{T}^V_{l^{-1}}) = (l^- \circ \mathcal{T}^V_{l^{-1}}) \bullet l = (k \mapsto l^{P(k)}) \bullet l$$

wobei P die Vorgängerfunktion auf den Ganzen Zahlen ist. Nach Definition von l^- gilt $l^{P(k)} \bullet l = l^k$, womit L wie gewünscht existiert. Damit können wir nun den Satz formulieren, aus dem 2.4.2 direkt durch Einsetzen von $*$ folgt:

Satz 2.4.3
Die Abbildungen sind „über S^1 punktweise quasiinvers“:

$$\prod_{x:S^1} \left(L(x) \circ W(x) \sim 1_{* =_{S^1} x}\right) \times \left(W(x) \circ L(x) \sim 1_{V(x)}\right)$$

Beweis Zunächst schreiben wir W_x statt $W(x)$ – und analog L_x.
Es ist $L_x(W_x(\gamma)) = \gamma$ für $\gamma : * =_{S^1} x$ zu zeigen. Erfreulicherweise lässt sich darauf nun direkt (=I) anwenden und es ist nur noch

$$r_{r_*} : L_*(W_*(r_*)) \equiv l^{\omega(r_*)} \equiv l^0 \equiv r_* = r_*$$

festzustellen – Fertig!
Die andere Richtung hätten wir auch schon in der spezielleren Version des Satzes nachrechnen können[5]:

$$W_*(L_*(k)) = \omega(l^k) = k$$

Trotzdem wollen wir natürlich per (S^1I) den Beweis abschließen. Dazu brauchen wir eine Gleichheit in $\mathcal{T}_l(k \mapsto r_k) = k \mapsto r_k$. Was auch immer die transportierte Abbildung sein mag – am Ende stehen rechts und links des Gleichheitszeichens Abbildungen mit Werten in Gleichheiten in $\mathbb{Z}$. Weil $\mathbb{Z}$ eine Menge ist, müssen diese gleich sein.

[5]Hier wird wieder „$\mathcal{T}_{\gamma \bullet \gamma'} = \mathcal{T}_{\gamma'} \circ \mathcal{T}_{\gamma}$“ benutzt.

Bibliography

[Gir72] J.Y. Girard. „Interprétation fonctionnelle et élimination des coupures dans l'arithmétique d'ordre supérieure". PhD thesis. Paris VII, 1972.

[LKV14] Peter Lumsdaine, Krzystof Kapulkin, and Vladimir Voevodsky. „The Simplicial Model of Univalent Foundations". In: (2014). arXiv: 1211.2851v2.

[Mat72] Per Matrin-Löf. *An intuitionistic theory of types*. 1972.

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