Vorlesungsskript zur Homotopietypentheorie

Felix Cherubini

23. August 2021

Inhaltsverzeichnis

Urteil	Bedeutung (evtl. im Kontext Γ)
$\Gamma \vdash t : A$	t ist ein Term vom Typ A
$\Gamma \vdash A$ Typ	A ist ein Typ
$\Gamma \vdash A \equiv B$	A und B sind (urteils-)gleiche Typen
$\Gamma \vdash t \equiv s : A$	t und s sind (urteils-)gleiche Terme des Typs A
Γ Kontext	Γ ist ein Kontext

Tabelle 1: Urteile

Dieses Skript entsteht als Begleitmaterial zur Vorlesung "Homotopietypentheorie" (HoTT), die ich im Sommersemester 2021 an der Universität Augsburg halte. Zweck dieses Skripts ist es, den Zuhörern und mir selbst zur Erinnerung an die Vorlesungsinhalte zu dienen – bei der Beschäftigung mit dem Thema ist es hilfreich in Lehrbücher zu schauen. Das sogenannte "HoTT-Book" ist sicher eine gute Quelle.

Wer Fehler findet und diese korrigieren oder darauf aufmerksam machen will, kann das auf der github-Seite dieses Skripts machen: https://github.com/felixwellen/HoTT-Vorlesung. Dort gibt es auch die Möglichkeit, sogenannte "Issues" anzulegen. Hier könnten sie etwa darüber informieren, wenn sie eine Passage nicht verstehen oder einen Fehler gefunden haben. Sie haben über sogenannte "Pull requests" die Möglichkeit, Fehler auch selbst zu korrigieren. Außer mir, Felix Cherubini, haben an der Entstehung des Skripts auch Daniel Albert und Lukas Stoll durch zahlreiche Korrekturen und Verbesserungen mitgewirkt.

Die Homotopietypentheorie ist eine eigenständige Art Mathematik zu betreiben und basiert auf einer abhängigen Typentheorie. Das bringt mit sich, dass wir zunächst erlernen werden, wie man in Homotopietypentheorie Objekte konstruiert, Aussagen formuliert und Beweise führt. Nach den Grundlagen werden wir uns in Richtung Homotopietheorie orientieren, einem Teilgebiet der Mathematik, in dem einige Vorzüge der Homotopietypentheorie zur Geltung kommen.

Im Folgenden werden wir nach und nach *Regeln* einführen (oder zumindest erwähnen), die schließlich zusammen eine Typentheorie ergeben. Diese werden wir noch um das sogenannte *Univalenzaxiom* erweitern.

Regeln

Wir werden zunächst nur Regeln kennenlernen, die Teil einer abhängigen Typentheorie sind. Regeln einer (abhängigen) Typentheorie können etwa so aussehen:

$$\frac{\Gamma \vdash f : A \to B \qquad \Gamma \vdash t : A}{\Gamma \vdash f(t) : B} \quad \text{(Beispiel)}$$

Über dem waagrechten Strich stehen die Voraussetzungen und darunter, was aus den Voraussetzungen geschlossen werden darf. Dabei kann man zum Beispiel "$\Gamma \vdash t : A$" lesen als "Im Kontext Γ gibt es einen Term t des Typs A". Wir werden nur anfangs mit Regeln arbeiten, um ein Verständnis für Typentheorie zu erlangen. Irgendwann werden wir Regeln wie die obige wieder, wie in der Mathematik üblich, sprachlich formulieren, etwa so

$$\text{Für } f : A \to B \text{ und } t : A \text{ gibt es ein } f(t) : B.$$

Wie wir später sehen werden, können diese Regeln zu Herleitungsbäumen kombiniert werden. Dieses Kombinieren ist der vollständig formale Weg, Beweise in der Homotopietypentheorie zu führen.

Ein Kontext wie "Γ" darf man sich als Liste von Variablen zusammen mit ihren Typen vorstellen. Also etwa so:

$$x_1 : A_1, \cdots, x_n : A_n$$

Dabei dürfen die Variablen nach ihrer Einführung verwendet werden. So könnte etwa in der Konstruktion des Typs A_2 die Variable x_1 verwendet werden. Dass der Kontext eine solche Liste ist, wird üblicherweise durch die strukturellen Regeln einer abhängigen Typentheorie festgelegt.

Ein Block der Form "$\Gamma \vdash t : A$" ist ein spezielles *Urteil*. Insgesamt gibt es die Urteile in Tabelle 1.

Eine Regel ist im Allgemeinen Fall nun von dieser Form:

$$\frac{\mathcal{U}_1 \qquad \dots \qquad \mathcal{U}_n}{\mathcal{U}_0} \quad \text{(Name)}$$

Für $n \in \mathbb{N} = \{0, 1, \dots\}$ und Urteile $\mathcal{U}_0, \dots, \mathcal{U}_n$.

Strukturregeln

Neben den Regeln für einzelne Typen, die wir in den folgenden Abschnitten kennenlernen, gibt es sogenannte *strukturelle Regeln* oder *Strukturregeln*. Diese legen fest, wie Grundsätzliches funktioniert, etwa wie Kontexte geformt werden dürfen und was man mit Gleichheitsurteilen anfangen darf. Hier ist ein Beispiel, die sogenannte "Weakening"-Regel oder *Abschwächungsregel*:

$$\frac{\Gamma \vdash \mathcal{U} \qquad \Gamma \vdash A \text{ Typ}}{\Gamma, x : A \vdash \mathcal{U}} \quad \text{(Weak)}$$

Das gilt für alles was man durch andere Regeln an der Stelle von $\mathcal{U}$ bekommen könnte. Die folgende *Variablenregel* erlaubt es Variablen aus dem Kontext zu benutzen:

$$\frac{\Gamma, x : A \text{ Kontext}}{\Gamma, x : A \vdash x : A} \quad \text{(Var)}$$

Die Regeln für die Urteilsgleichheit legen fest, dass diese eine Äquivalenzrelation auf Termen und Typen ist. Außerdem gibt es Strukturregeln und sogenannte *Kongruenzregeln*, die es letztendlich erlauben, Terme und Typen durch Urteilsgleiches zu ersetzen. Wir erlauben es uns einfach zu Ersetzen und führen diese Regeln nicht aus.

Weiter sind Typentheorien typischerweise so aufgebaut, dass wenn es möglich ist $t : A$ herzuleiten, es auch immer möglich ist, A Typ herzuleiten, in welchem Fall es wieder möglich ist, herzuleiten, dass der Kontext in dem das gilt ein Kontext ist. Allgemeiner erlauben wir uns stets notwendige Voraussetzungen für vorliegende Urteile zu verwenden. Also etwa der Schluss von einem Urteil, in dem der Typ "$A \to B$" vorkommt, zum Urteil "A Typ". Wir wollen diese Tatsachen frei verwenden und wie Regeln einsetzen, die wir zusammenfassend mit "Str" bezeichnen.

Diese Regeln führen wir hier nicht auf, eine gute Quelle, um die genauen Regeln im Bedarfsfall anzuschauen, ist das (noch nicht erschienene) Lehrbuch von Egbert Rijke oder Anhang A.2 des HoTT-Books.

1 Abhängige Typentheorie

1.1 Abhängige Typen und Abhängige Produkte

Von einem *Abhängigen Typen* spricht man im Fall eine Urteils

$$\Gamma, x : A \vdash B(x) \text{ Typ}$$

Der Kontext besteht also aus mindestens einer Variablen x, die im abhängigen Typen $B(x)$ vorkommen darf. Wir verwenden die Schreibweise "$B(x)$" um die Abhängigkeit klar zu machen – in der Typentheorie ist es eher üblich hier nur "B" zu schreiben. Ein Beispiel, das wir später mit unserer Typentheorie konstruieren könnten, ist der Typ der Listen der Länge n, wobei n eine Variable des Typs $\mathbb{N}$ ist.

Eine wichtige Besonderheit der abhängigen Typentheorie ist es, dass der Typ der Werte einer Funktion variieren darf. Diese allgemeineren Funktionen sind in sogenannten *abhängigen Produkten* oder *abhängigen Funktionstypen* enthalten. Die folgende Regel erlaubt es uns, diesen Typ zu formen:

$$\frac{\Gamma, x : A \vdash B(x) \text{ Typ}}{\Gamma \vdash \prod_{x:A} B(x) \text{ Typ}} \quad \text{(ΠF)}$$

Abhängige Funktionen, die Elemente des abhängigen Produkts, können mit dieser Regel konstruiert werden:

$$\frac{\Gamma, x : A \vdash t(x) : B(x)}{\Gamma \vdash x \mapsto t(x) : \prod_{x:A} B(x)} \quad \text{(Π)}$$

Der Term "$x \mapsto t(x)$" wird in der Typentheorie und allgemeiner in der Informatik geschrieben als "$\lambda x.t(x)$". Wir schreiben auch manchmal $(x : A) \mapsto t(x)$, wenn der Typ der Variablen nicht klar ist. Auch bei abhängigen Termen ist es in der Typentheorie eigentlich üblich nur "$t : B$" zu schreiben. Das

zusätzliche "(x)" steht hier nur zum besseren Verständnis durch Ähnlichkeit zu mathematischen Konventionen. Die nächste Regel erlaubt es uns, abhängige Funktionen anzuwenden:

$$\frac{\Gamma \vdash f : \prod_{x:A} B(x) \qquad \Gamma \vdash a : A}{\Gamma \vdash f(a) : B(a)} \quad (\Pi\mathrm{E})$$

Damit ist allerdings noch nicht gesagt, wie das Einsetzen eines Wertes in eine Funktion funktioniert. Dafür müssen alle Vorkommen einer Variablen in einem Funktionsterm durch den eingesetzten Term ersetzt werden. Diesen Vorgang nennt man Substitution und wir verwenden dafür die Notation $t(a)$ statt dem in der Informatik üblichen "$t[a/x]$". Die folgende Regel sagt uns, dass wir durch Einsetzen von $a : A$ den Wert der Funktion $(x : A) \mapsto t(x)$, berechnen dürfen:

$$\frac{\Gamma, x : A \vdash t(x) : B(x) \qquad \Gamma \vdash a : A}{\Gamma \vdash (x \mapsto t(x))(a) \equiv t(a) : B(a)} \quad (\Pi\beta)$$

Eine Besonderheit der abhängigen Produkte ist es, dass man folgende Urteilsgleichheit noch zusätzlich fordert:

$$\frac{\Gamma \vdash f : \prod_{x:A} B(x)}{\Gamma \vdash f \equiv (x \mapsto f(x)) : \prod_{x:A} B(x)} \quad (\Pi\eta)$$

Für alle weitere Typen, die wir einführen werden, wird sich ein ähnliches Schema ergeben. Es gibt stets eine Regel, die es erlaubt den Typ zu formen, eine oder mehrere für die Konstruktion von Elementen und wieder eine oder mehrere, die festlegen, wie die Elemente des Typs verwendet werden dürfen.

Wenn $B(x)$ eigentlich gar nicht von $x : A$ abhängt, also x nicht in B vorkommt, kann der Typ der abhängigen Funktionen, $\prod_{x:A} B(x)$, spezialisiert werden zum üblichen Funktionstyp $A \to B$:

Damit haben wir für beliebige Typen A, B in einem Kontext Γ den Typ der Funktionen von A nach B:

Definition 1.1.1

(a) Für Typen A und B gibt es stets den *Typ der Funktionen* $A \to B$, der mit der folgenden Herleitung geformt werden kann:

$$\frac{\dfrac{\Gamma \vdash B \text{ Typ} \qquad \Gamma \vdash A \text{ Typ}}{\Gamma, x : A \vdash B \text{ Typ}} \text{ (Weak)}}{\Gamma \vdash A \to B \text{ Typ}} \quad (\Pi\mathrm{F})$$

Die Terme dieses Typs nennen wir *Funktionen.*

(b) Unter der *Identität* id_A auf einem Typ A verstehen wir die folgende Konstruktion:

$$\frac{\dfrac{\dfrac{\dfrac{\Gamma \vdash A \text{ Typ} \qquad \Gamma \vdash A \text{ Typ}}{\Gamma, x : A \vdash A \text{ Typ}} \text{ (Weak)}}{\Gamma, x : A \text{ Kontext}} \text{ (Str)}}{\Gamma, x : A \vdash x : A} \text{ (Var)}}{\Gamma \vdash x \mapsto x : A \to A} \quad (\mathrm{III})$$

Zusammengefasst: $\mathrm{id}_A :\equiv (x : A) \mapsto x$.

(c) Für Typen A, B, C und Funktionen $f : A \to B$, $g : B \to C$ bezeichnen wir mit $g \circ f$ deren *Komposition.* Ganz genau ist die Komposition der durch die folgende Herleitung gegebene Term:

$$\frac{\dfrac{\dfrac{\Gamma \vdash g : B \to C \qquad \Gamma \vdash A \text{ Typ}}{\Gamma, x : A \vdash g : B \to C} \text{ (Weak)} \qquad \dfrac{\Gamma \vdash f : A \to B \qquad \dfrac{\dfrac{\Gamma \vdash f : A \to B}{\Gamma \vdash A \text{ Typ}} \text{ (Str)}}{\Gamma, x : A \vdash x : A} \text{ (Weak, Var)}}{\Gamma, x : A \vdash f(x) : B} \text{ (}\Pi\text{E)}}{\Gamma, x : A \vdash g(f(x)) : C} \text{ (}\Pi\text{E)}}{\Gamma \vdash x \mapsto g(f(x)) : A \to C} \quad (\mathrm{III})$$

In Zukunft werden wir solche Sachverhalte auch einfach durch "$f \circ g :\equiv x \mapsto f(g(x))$" ausdrücken.

Bemerkung 1.1.2

(a) Für jeden Typ A und $x : A$ gilt $\mathrm{id}_A(x) \equiv x$.

(b) Für Typen A, B und $f : A \to B$ gilt: $f \circ \mathrm{id}_A \equiv f$ und $\mathrm{id}_B \circ f \equiv f$.

1.2 Natürliche Zahlen

Im Gegensatz zum abhängigen Produkt ist der Typ der *Natürlichen Zahlen* nicht von anderen Typen abhängig. Dementsprechend ist die Formierungsregel etwas einfacher:

$$\frac{\Gamma \text{ Kontext}}{\Gamma \vdash \mathbb{N} \text{ Typ}} \quad (\mathbb{N}\text{F})$$

Eine Neuheit ist, dass es zwei Regeln für die Konstruktion von Termen gibt:

$$\frac{\Gamma \text{ Kontext}}{\Gamma \vdash 0_{\mathbb{N}} : \mathbb{N}} \quad (\mathbb{N}\text{I1}) \qquad \frac{\Gamma \vdash n : \mathbb{N}}{\Gamma \vdash \mathrm{succ}_{\mathbb{N}}(n) : \mathbb{N}} \quad (\mathbb{N}\text{I2})$$

Soweit heißt das nur, dass man stets die Natürlichen Zahlen verwenden darf, es ein Element $0_{\mathbb{N}}$ und zu jeder natürlichen Zahl einen Nachfolger gibt. Den Index "$_{\mathbb{N}}$" soll Verwechslungen verhindern und wird gelegentlich wegegelassen.

Definition 1.2.1

Wir verwenden die übliche Schreibweise für natürliche Zahlen:

$$0 :\equiv 0_{\mathbb{N}}, 1 :\equiv \mathrm{succ}_{\mathbb{N}}(0), 2 :\equiv \mathrm{succ}_{\mathbb{N}}(1), \ldots$$

Die nächste Regel wird es uns erlauben, per Induktion Aussagen über die natürlichen Zahlen zu zeigen, aber auch Funktionen auf den natürlichen Zahlen zu konstruieren:

$$\frac{\Gamma, n : \mathbb{N} \vdash P(n) \text{ Typ} \qquad \Gamma \vdash \mathbf{IA} : P(0) \qquad \Gamma, n : \mathbb{N}, \mathbf{IH} : P(n) \vdash \mathbf{IS}(n, \mathbf{IH}) : P(\mathrm{succ}(n))}{\Gamma \vdash \mathrm{ind}_{\mathbb{N}}(P, \mathbf{IA}, \mathbf{IS}) : \prod_{n:\mathbb{N}} P(n)} \quad (\mathbb{N}\text{E})$$

Um die Regel mit bekannten Vorstellungen von Induktion zusammenzubringen, stellt man sich $P(n)$ als Aussage über die Zahl n vor. Wenn man es nun schafft, einen Term $p : P(n)$ zu konstruieren, bedeutet das, dass man die Aussage $P(n)$ bewiesen hat. In dieser Lesart ist das abhängige Produkt $\prod_{n:\mathbb{N}} P(n)$ nichts anderes als "$\forall n \in \mathbb{N}$ gilt $P(n)$". Nun muss man, um per Induktion eine Aussage zu zeigen, den Induktionsanfang (**IA**) zeigen und den Induktionsschritt (**IS**) aus der Induktionshypothese (**IH**) folgern. Die zu diesen Einzelteilen passenden Terme sind in der Regel oben entsprechend benannt.

Bemerkung 1.2.2

Wir werden später die Möglichkeit haben, mittels abhängigen Typen Aussagen über natürliche Zahlen zu konstruieren, wie z.B.

$$P(n) :\equiv \text{ "}n \cdot (n+1) = 2 \cdot (n + (n-1) + \cdots + 1)\text{"}$$

Dazu fehlen uns momentan allerdings noch Typen für die zweite Art von Gleichheit "=". Bis wir diese einführen können, müssen wir uns noch mit konstanter Abhängigkeit begnügen.

Der wichtige Spezialfall der Regel $\mathbb{N}$E für (nicht-abhängige) Funktionen, heißt Rekursion:

Definition 1.2.3

Sei A ein Typ. Dann ist für $f_0 : A$ und $f_s : \mathbb{N} \to (A \to A)$ durch $\mathbb{N}$E eine Funktion

$$\mathrm{rec}_{\mathbb{N}}(A, f_0, f_s) :\equiv \mathrm{ind}_{\mathbb{N}}(n : \mathbb{N} \vdash A, f_0, f_s) : \mathbb{N} \to A$$

gegeben. Diese Art Funktionen (auf $\mathbb{N}$) zu definieren nennt man ($\mathbb{N}$-)*Rekursion*.

Beispiel 1.2.4

Sei $d : \mathbb{N} \to \mathbb{N}$ gegeben durch

$$d :\equiv \mathrm{rec}_{\mathbb{N}}(\mathbb{N}, 0, n \mapsto (k \mapsto \mathrm{succ}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(k))))$$

das entspricht einer rekursiven Definition durch die Gleichungen

$$d(0) :\equiv 0$$
$$d(n+1) :\equiv d(n) + 2$$

– also einer Funktion, die ihr Argument verdoppelt.

Noch haben wir keine Möglichkeit, eine durch Rekursion oder Induktion definierte Funktion für ein Argument auszuwerten. Dazu brauchen wir β-Regeln, die nichts anderes sagen, als dass eine durch Induktion definierte Funktion, die durch Induktionsanfang und Schritt gegebenen Werte auch annimmt. Es sollen also für mit $\mathbb{N}$E definierte Funktionen gelten:

$$\mathrm{ind}_{\mathbb{N}}(P, \mathrm{IA}, \mathrm{IS})(0_{\mathbb{N}}) \equiv \mathrm{IA}(\text{Für } n : \mathbb{N})\ \mathrm{ind}_{\mathbb{N}}(P, \mathrm{IA}, \mathrm{IS})(\mathrm{succ}_{\mathbb{N}}(n)) \equiv \mathrm{IS}(n, \mathrm{ind}_{\mathbb{N}}(P, \mathrm{IA}, \mathrm{IS})(n))$$

und damit für die Rekursion mit den Bezeichnern aus Definition 1.2.3:

$$\mathrm{rec}_{\mathbb{N}}(A, f_0, f_s)(0) \equiv f_0 \mathrm{rec}_{\mathbb{N}}(A, f_0, f_s)(\mathrm{succ}_{\mathbb{N}}(n)) \equiv f_s(n, \mathrm{rec}_{\mathbb{N}}(A, f_0, f_s)(n))$$

Die vollständigen β-Regeln sind wie folgt:

$$\frac{\Gamma, n : \mathbb{N} \vdash P(n)\ \mathrm{Typ} \qquad \Gamma \vdash \mathrm{IA} : P(0) \qquad \Gamma, n : \mathbb{N}, \mathrm{IH} : P(n) \vdash \mathrm{IS}(n, \mathrm{IH}) : P(\mathrm{succ}(n))}{\Gamma \vdash \mathrm{ind}_{\mathbb{N}}(P, \mathrm{IA}, \mathrm{IS})(0_{\mathbb{N}}) \equiv \mathrm{IA} : P(0_{\mathbb{N}})}$$

$$(\mathbb{N}\beta_1)\frac{\Gamma, n : \mathbb{N} \vdash P \qquad \Gamma, n : \mathbb{N}, \mathrm{IH} : P(n) \vdash}{\Gamma \vdash \mathrm{ind}_{\mathbb{N}}(P, \mathrm{IA}, \mathrm{IS})(\mathrm{succ}_{\mathbb{N}}(k)}$$

Damit können wir Werte der Funktion aus Beispiel 1.2.4 berechnen:

Beispiel 1.2.5
Für die Funktion $d : \mathbb{N} \to \mathbb{N}$ aus Beispiel 1.2.4 ergibt sich mit den β-Regeln nun etwa:

$$\begin{aligned}
d(3) &\equiv d(\mathrm{succ}_{\mathbb{N}}(2)) \\
&\equiv (k \mapsto \mathrm{succ}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(k)))(d(2)) && (\mathbb{N}\beta_2) \\
&\equiv \mathrm{succ}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(d(2))) && (\Pi\beta) \\
&\equiv \mathrm{succ}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(d(1))))) && (\mathbb{N}\beta_2, \Pi\beta) \\
&\equiv \mathrm{succ}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(d(0)) && (\mathbb{N}\beta_2, \Pi\beta) \\
&\equiv 6 && (\mathbb{N}\beta_1)
\end{aligned}$$

Folgende Konvention werden wir ab jetzt verwenden:

Konvention 1.2.6
(a) Wir verwenden die folgenden Klammerungen:

$$\prod_{x:A} B \to C \coloneqq \prod_{x:A} (B \to C)$$

und auch im Allgemeinen, dass $\prod$ als letzter Typ-Former ausgeführt wird.

(b) Weiter klammern wir iterierte Funktionen von rechts:

$$A_1 \to A_2 \to \cdots \to A_n \coloneqq A_1 \to (A_2 \to (\dots (A_{n-1} \to A_n) \dots)$$

(c) Wir erlauben uns etwas Freiheit beim Schreiben von Funktionsanwendungen, also etwa für $f : A \to B \to C$ auch mal $f(a, b)$ statt $f(a)(b)$ zu schreiben oder auch afb, wenn f ein Operator ist.

Zum Abschluss unserer ersten Betrachtung der natürlichen Zahlen werden wir nun die arithmetischen Operationen definieren:

Definition 1.2.7
(a) Die Addition $+ : \mathbb{N} \to \mathbb{N} \to \mathbb{N}$ ist gegeben durch die folgenden Urteilsgleichungen:

$$\begin{aligned}
0 + k &\coloneqq k \\
\mathrm{succ}_{\mathbb{N}}(n) + k &\coloneqq \mathrm{succ}_{\mathbb{N}}(n + k)
\end{aligned}$$

Formal ist $+$ gegeben durch:

$$+ \coloneqq \mathrm{rec}_{\mathbb{N}}(\mathbb{N} \to \mathbb{N}, k \mapsto k, n \mapsto f \mapsto (k \mapsto \mathrm{succ}_{\mathbb{N}}(f(k))))$$

(b) Die Multiplikation $\cdot : \mathbb{N} \to \mathbb{N} \to \mathbb{N}$ ist gegeben durch:

$$\begin{aligned}
0 \cdot k &\coloneqq 0 \\
\mathrm{succ}_{\mathbb{N}}(n) \cdot k &\coloneqq (n \cdot k) + k
\end{aligned}$$

Bzw.:

$$\cdot \coloneqq \mathrm{rec}_{\mathbb{N}}(\mathbb{N} \to \mathbb{N}, k \mapsto 0, n \mapsto f \mapsto (k \mapsto f(k) + k))$$

Beispiel 1.2.8
Wir können nun wie folgt mit natürlichen Zahlen rechnen:

$$\begin{aligned}
1+1 &\equiv \mathrm{succ}_{\mathbb{N}}(0)+1\\
&\equiv +(\mathrm{succ}_{\mathbb{N}}(0))(1)\\
&\equiv (n \mapsto f \mapsto (k \mapsto \mathrm{succ}_{\mathbb{N}}(f(k))))(0)(+(0))(1)\\
&\equiv (f \mapsto (k \mapsto \mathrm{succ}_{\mathbb{N}}(f(k))))(l \mapsto l)(1)\\
&\equiv (k \mapsto \mathrm{succ}_{\mathbb{N}}((l \mapsto l)(k)))(1)\\
&\equiv (k \mapsto \mathrm{succ}_{\mathbb{N}}(k))(1)\\
&\equiv 2
\end{aligned}$$

1.3 Induktive Typen

Der Typ der natürlichen Zahlen ist ein Beispiel für einen sogenannten *induktiven Typ*. Wir werden in diesem Abschnitt ein paar weitere Typen dieser Bauart kennenlernen. Das sich dabei wiederholende Muster ist, dass der Typ jeweils im wesentlichen durch seine Einführungsregeln gegeben ist. Eine Einführungsregel besteht im Wesentlichen aus einer Funktion in den Induktiven Typ, oder genauer ihrer Signatur. Diese Funktionen werden wir von nun an *Konstruktoren* nennen. Im Fall der natürlichen Zahlen gab es die beiden Konstruktoren $0_{\mathbb{N}} : \mathbb{N}$ und $\mathrm{succ}_{\mathbb{N}} : \mathbb{N} \to \mathbb{N}$.

Wir beginnen mit einem Typen, der in noch zu klärendem Sinn genau einen Term hat.

Regeln 1.3.1
Der *Einheitstyp* $\mathbf{1}$ ist der Induktive Typ mit Konstruktur $* : \mathbf{1}$. Dadurch ergibt sich der folgende Satz von Regeln:

$$\frac{\Gamma \text{ Kontext}}{\Gamma \vdash \mathbf{1} \text{ Typ}} \quad (\mathbf{1}\mathrm{F}) \qquad \frac{\Gamma \text{ Kontext}}{\Gamma \vdash * : \mathbf{1}} \quad (\mathbf{1}\mathrm{I}) \qquad \frac{\Gamma, x : \mathbf{1} \vdash P(x) \text{ Typ} \qquad \Gamma \vdash p : P(*)}{\Gamma \vdash \mathrm{ind}_{\mathbf{1}}(P,p) : \prod_{x:\mathbf{1}} P(x)} \quad (\mathbf{1}\mathrm{E}) \qquad \frac{\Gamma, x : \mathbf{1} \vdash P(x) \text{ Typ} \qquad \Gamma \vdash p : P(}{\Gamma \vdash \mathrm{ind}_{\mathbf{1}}(P,p)(*) \equiv p : P(*)}$$

Beispiel 1.3.2
Wir können die beiden Funktionen

$$(x \mapsto 0) : \mathbf{1} \to \mathbb{N} \text{ und } \mathrm{rec}_{\mathbf{1}}(\mathbb{N}, 0) : \mathbf{1} \to \mathbb{N}$$

definieren. Es ist allerdings nicht möglich zu zeigen, dass $(x \mapsto 0) \equiv \mathrm{rec}_{\mathbf{1}}(\mathbb{N}, 0)$ gilt. Später werden wir in der Lage sein (mittels Induktion) zu zeigen, dass Objektgleichheit "=" zwischen diesen Funktionen gilt.

Regeln 1.3.3
Der *leere Typ* $\emptyset$ ist der Induktive Typ ohne Konstruktur. Dadurch ergibt sich der folgende Satz von Regeln:

$$\frac{\Gamma \text{ Kontext}}{\Gamma \vdash \emptyset \text{ Typ}} \quad (\emptyset\mathrm{F}) \qquad \frac{\Gamma, x : \emptyset \vdash P(x) \text{ Typ}}{\Gamma \vdash \mathrm{ind}_{\emptyset}(P) : \prod_{x:\emptyset} P(x)} \quad (\emptyset\mathrm{E})$$

Regeln 1.3.4
Der *zweielementige Typ* oder *Bool* $\mathbf{2}$ ist ein Induktiver Typ mit den zwei Konstruktoren

$$0_{\mathbf{2}} : \mathbf{2} \qquad 1_{\mathbf{2}} : \mathbf{2}$$

Wir verzichten diesmal auf Angabe der Regeln.

Es ist auch möglich, einen Induktiven Typen zu definieren, der von einem oder mehreren *Parametertypen* abhängt. Der folgende induktive Typ kann für je zwei Typen geformt werden und ist typentheoretische Version der disjunkten Vereinigung:

Regeln 1.3.5
Das *Koprodukt* zweier Typen A und B ist der induktive Typ $A \sqcup B$ mit den Konstruktoren

$$\iota_1 : A \to A \sqcup B \qquad \iota_2 : B \to A \sqcup B$$

Eine Funktion $f : A \sqcup B \to C$ in einen Typen C, kann also definiert werden durch Angabe zweier Funktionen $f_1 : A \to C$ und $f_2 : B \to C$.

Beispiel 1.3.6

(a) Das Koprodukt erlaubt eine Alternative Konstruktion des Typs $\mathbf{2}$ als $\mathbf{1} \sqcup \mathbf{1}$. Wir können zwar noch nicht ausdrücken, dass zwei Typen gleich sind (abgesehen von der Urteilsgleichheit, die uns hier nicht helfen würde), wollen aber trotzdem schonmal die beiden Abbildungen definieren, die uns das später erlauben werden:

$$f(0_{\mathbf{2}}) :\equiv \iota_1(*) \qquad g(\iota_1(*)) :\equiv 0_{\mathbf{2}}$$
$$f(1_{\mathbf{2}}) :\equiv \iota_2(*) \qquad g(\iota_2(*)) :\equiv 1_{\mathbf{2}}$$

(b) Für Typen A, B gibt es stets die Abbildung

$$\mathrm{ind}_{\sqcup}(\mathbf{2}, (a : A) \mapsto 0_{\mathbf{2}}, (b : B) \mapsto 1_{\mathbf{2}}) : A \sqcup B \to \mathbf{2}.$$

1.4 Gleichheit

Da wir uns im Folgenden der Situation nähern die Typentheorie einsetzen zu können, um über mathematische Objekte zu reden, wollen wir auch weniger syntaktische Sprechweisen bevorzugen und etwa eher von *Elementen eines Typs* statt Termen sprechen. Außerdem wollen wir es uns nun erlauben, verschachtelte $\prod$-Ausdrücke über den gleichen Typen wie folgt abzukürzen:

$$\prod_{x:A} \prod_{y:A} \cdots :\equiv \prod_{x,y:A} \cdots$$

In diesem Abschnitt werden wir die *Gleichheit von Objekten* $x = y$ einführen. Diese wollen wir im Folgenden auch einfach nur *Gleichheit* nennen. Im Gegensatz zur klassischen Mathematik ist die Gleichheit zwischen zwei Elementen $x, y : A$ eines Typs selbst wieder ein vollwertiger Typ $x =_A y$. Wir werden auch tatsächlich Beispiele von Typen sehen, deren Gleichheitstypen mehrere verschiedene Elemente enthalten. Dies kann man sich als die Neuheit vorstellen, dass Dinge auf mehrere Arten gleich sein können.

Regeln 1.4.1

Für zwei Elemente x, y eines Typs A können wir den Typ $x =_A y$ der *Gleichheiten* zwischen x und y formen. Dieser wird auch *Identitätstyp* genannt und ist als induktiver Typ durch den Konstruktor

$$\mathrm{refl} : \prod_{x:A} x =_A x$$

festgelegt. Wir nehmen von nun an die sich daraus ergebenden Regeln an, die wir im Folgenden diskutieren werden.

Als Formierungs und Einführungsregeln ergeben sich:

$$\frac{\Gamma \vdash x : A \qquad \Gamma \vdash y : A}{\Gamma \vdash x =_A y \ \mathrm{Typ}} \ (=\mathrm{F}) \qquad \frac{\Gamma \vdash x : A}{\Gamma \vdash \mathrm{refl}_x : x =_A x} \ (=\mathrm{I})$$

Bevor wir weitere Regeln diskutieren, machen wir zunächst Beispiele:

Beispiel 1.4.2

(a) Für $x, y : \mathbf{1}$ können wir nun den Typ $x =_{\mathbf{1}} y$ formen. Weiter können wir mit $\mathrm{ind}_{\mathbf{1}}$ auch recht deutlich beschreiben, wie dieser Typ aussieht:

$$\mathrm{ind}_{\mathbf{1}}(x : \mathbf{1} \vdash x =_{\mathbf{1}} *, \mathrm{refl}_*) : \prod_{x:\mathbf{1}} x = *$$

(b) Für den leeren Typ können wir zeigen, dass je zwei $x, y : \emptyset$ gleich sind:

$$\mathrm{ind}_{\emptyset}(x : \emptyset \vdash \prod_{y:\emptyset} x = y) : \prod_{x,y:\emptyset} x = y$$

(c) Wenig überraschend, ist jedes $x : \mathbf{2}$ entweder gleich $0_{\mathbf{2}}$ oder $1_{\mathbf{2}}$, was wir mit dem Koprodukt ausdrücken können:

$$\mathrm{ind}_{\mathbf{2}}(x : \mathbf{2} \vdash (x = 0_{\mathbf{2}}) \sqcup (x = 1_{\mathbf{2}}), \iota_1(\mathrm{refl}_{0_{\mathbf{2}}}), \iota_2(\mathrm{refl}_{1_{\mathbf{2}}})) : \prod_{x:\mathbf{2}} (x = 0_{\mathbf{2}}) \sqcup (x = 1_{\mathbf{2}})$$

Die Eliminationsregel, bzw die *Induktion für Gleichheit*, die auch *Pfadinduktion* genannt wird, stellt sich als erstaunlich vielseitig heraus. Sie besagt, dass für einen abhängigen Typen über dem Gleichheitstyp, also etwa $x : A, y : A, p : x =_A y \vdash B(p)$ und eine Vorgabe für refl_x also ein Element $b_r : \prod_{x:A} B(\mathrm{refl}_x)$ bereits eine abhängige Funktion wie folgt gegeben ist:

$$\mathrm{ind}_=(B, b_r) : \prod_{x,y:A} \prod_{p:x=_A y} B(p)$$

Eine wünschenswerte Eigenschaft für Gleichheitsbegriffe ist, dass es sich um Äquivalenzrelationen handelt. Da es sich bei unserem Gleichheitstyp im Allgemeinen um mehr als eine Relation handelt, haben statt den Eigenschaften Reflexivität, Symmetrie und Transitivität eine sogenannte *Gruppoidstruktur*[1]. Das bedeutet, dass diese drei Eigenschaften zu Operationen verallgemeinert werden. Die Reflexivität ist bereits durch den Konstruktor gegeben und erlaubt es uns für jedes $x : X$ eine Gleichheit $x =_X x$ zu *konstruieren.* Statt der Symmetrie haben wir eine Umkehroperation oder Inversionsoperation von $x = y$ nach $y = x$ und die Transitivität wird zu einer Verkettungsoperation oder Konkatenation von Gleichheiten:

Definition 1.4.3

(a) Für $p : x =_A y$ bezeichnen wir mit $p^{-1} : y =_A x$ die *inverse Gleichheit*, also eine Funktion $\_^{-1} : x =_A y \to y =_A x$, welche wir durch Vorgabe für "refl_x" definieren können:

$$(\mathrm{refl}_x)^{-1} :\equiv \mathrm{refl}_x$$

bzw durch die folgende Induktion:

$$\mathrm{ind}_=(x, y : A, p : x =_A y \vdash y =_A x; \mathrm{refl}_x) : \prod_{x,y:A} \prod_{p:x=_A y} y =_A x$$

(b) Seien $x, y, z : A$. Für zwei Gleichheiten $p : x =_A y$ und $q : y =_A z$ ist die *Konkatenation* $p{\cdot}q$ gegeben durch:

$$\mathrm{refl}_x{\cdot}q :\equiv q$$

Der Induktionsterm sieht für $z : A$ wie folgt aus:

$$\mathrm{ind}_=(x, y : A, p : x =_A y \vdash y =_A z \to x =_A z; \mathrm{id}_{x=_A z}) : \prod_{x,y:A} (x =_A y) \to (y =_A z) \to (x =_A z)$$

Beispiel 1.4.4

Nach Beispiel 1.4.2 gibt es einen Term

$$\prod_{x:\mathbf{1}} x = *$$

Damit können wir nun auch zeigen, dass alles in $\mathbf{1}$ paarweise gleich ist:

$$(x, y : \mathbf{1}) \mapsto k(x){\cdot}k(y)^{-1} : \prod_{x,y:\mathbf{1}} x = y$$

Nach diesen Konstruktionen wirkt die Induktionsregel für Gleichheit vielleicht etwas zu stark. Tatsächlich wenden wir die nötige Arbeit, um zu Beweisen zu kommen, etwas versteckt beim Definieren der abhängigen Typen auf, auf die wir die Induktionsregel anwenden. Dass es dabei etwas zu beachten gibt, sieht man an den Grenzen der Induktionsregel:

Bemerkung 1.4.5

Das folgende kann nicht mit Induktion (und auch sonst mit keiner Regel der Vorlesung) gezeigt werden[2]:

$$\prod_{x:A} \prod_{p:x=_A x} p =_{x=x} \mathrm{refl}_x$$

[1] Die Gleichungen, die in einem Gruppoid gelten würden, gelten hier nicht strikt.

[2] So etwas lässt sich nicht so einfach formal zeigen.

Das Problem dabei ist, dass wir keine Möglichkeit haben zu fordern, dass die Endpunkte x und y die gleiche Variable sind. Der abhängige Typ $x : A, p : x =_A x \vdash p = \mathrm{refl}_x$ kann also nicht in die Form gebracht werden, die wir für die Induktionsregel brauchen. In der Anschauung bedeutet das, dass es wichtig ist, dass wir Wege mit frei-beweglichen Endpunkten haben.

Die Notation der oben definierten Operationen erinnert stark an die einer Gruppe. Tatsächlich können wir zeigen, dass auch ähnliche Gesetze für die Gleichheit gelten. Zunächst verhält sich refl_x ähnlich wie ein Neutralelement:

Bemerkung 1.4.6
Seien A ein Typ, $x, y : A$ und $p : x =_A y$. Dann gelten:
(a) $\mathrm{refl}_x \cdot p = p$

(b) $p \cdot \mathrm{refl}_y = p$

Beweis (a) Diese Gleichung gilt bereits urteilsmäßig.

(b) Unser Ziel ist:

$$\prod_{x,y:A} \prod_{p:x=_A y} p \cdot \mathrm{refl}_y = p$$

Mit Induktion reicht es also zu zeigen

$$\prod_{x:A} \mathrm{refl}_x \cdot \mathrm{refl}_x = \mathrm{refl}_x$$

Der Term $\mathrm{refl}_x \cdot \mathrm{refl}_x$ ist aber bereits nach Definition urteilsmäßig gleich refl_x. Also ist $x \mapsto \mathrm{refl}_{\mathrm{refl}_x}$ ein passender Term. □

Wir wollen nun damit fortfahren, den Namen *Inversion* zu rechtfertigen. Anschaulich, ist für eine Gleichheit $p : x = y$ ihr Inverses $p^{-1} : y = x$ einfach diejenige Gleichheit, die p rückwärts durchläuft. Der Weg $p \cdot p^{-1}$ verläuft also von x nach y und dann auf dem gleichen Weg wieder zurück. Insgesamt lässt sich der Weg $p \cdot p^{-1}$ in Richtung x zusammenziehen zum konstanten Weg $\mathrm{refl}_x : x = x$.

Formal können wir diese Tatsache mit einer Gleichheitsinduktion fassen:

Bemerkung 1.4.7
Für einen Typ A und Elemente $x, y : A$ gilt für jedes $p : x =_A y$:

$$p \cdot p^{-1} =_{x=_A x} \mathrm{refl}_x$$

Beweis Wir wollen also etwas in folgendem Typ konstruieren:

$$\prod_{x,y:A} \prod_{p:x=y} p \cdot p^{-1} = \mathrm{refl}_x.$$

Um das mit Induktion zu erledigen, müssen wir das für $p \equiv \mathrm{refl}_x$ zeigen, also einen Term von

$$\prod_{x:A} \mathrm{refl}_x \cdot \mathrm{refl}_x^{-1} = \mathrm{refl}_x$$

angeben. Nach Definition von $\_^{-1}$ gilt $\mathrm{refl}_x^{-1} \equiv \mathrm{refl}_x$, also müssen wir für $x : A$ eigentlich nur etwas in $\mathrm{refl}_x \cdot \mathrm{refl}_x = \mathrm{refl}_x$ konstruieren. Nach Definition von $\_\cdot\_$ ist das aber nur $\mathrm{refl}_x = \mathrm{refl}_x$. D.h. wir haben mit

$$(x : A) \mapsto \mathrm{refl}_{\mathrm{refl}_x}$$

den gesuchten Term konstruiert. □

Bei Beweisen dieser Art ist es wichtig, dass es sich eigentlich um Konstruktionen handelt. D.h. wir konstruieren Terme, die wir später auch verwenden werden und es kann passieren, dass die spezielle Konstruktion eine Rolle spielt. In einem Typ A kann es auch durchaus mehrere verschiedene Gleichheiten in $\mathrm{refl}_x \cdot p = p$ geben.

Als letzte elementare Gleichung für das Rechnen mit Gleichheiten, zeigen wir die Assoziativität von $\_\cdot\_$.

Bemerkung 1.4.8
Für einen Typ A und $x, y, z, w : A$ gilt:

$$\prod_{p:x=y} \prod_{q:y=z} \prod_{r:z=w} (p \cdot q) \cdot r = p \cdot (q \cdot r)$$

Beweis Durch Umsortieren der Abhängigkeiten, was wir nach den Strukturregeln dürfen, können wir Induktion auf den folgenden abhängigen Typen anwenden:

$$x : A, y : A, p : x =_A y \vdash \prod_{z,w:A} \prod_{q:y=z} \prod_{r:z=w} (p \cdot q) \cdot r = p \cdot (q \cdot r)$$

Also müssen wir nur noch folgendes zeigen:

$$\prod_{x:A} \prod_{z,w:A} \prod_{q:x=z} \prod_{r:z=w} (\mathrm{refl}_x \cdot q) \cdot r = \mathrm{refl}_x \cdot (q \cdot r)$$

Aber die Gleichung lässt sich nun mit per Definition gegebenen Urteilsgleichheiten zu $q \cdot r = q \cdot r$ reduzieren, was durch $\mathrm{refl}_{q \cdot r}$ gegeben ist. □

Bemerkung 1.4.9
Für einen beliebigen Typen A und ein Element $x : A$ erfüllt der Typ $x =_A x$ die naheliegenden Übersetzungen der Axiome einer Gruppe. Allerdings werden wir später noch mehr fordern, damit wir einen Typ eine Gruppe nennen.

Auch die Assoziativität ist eine *spezielle* Gleichheit in $(p \cdot q) \cdot r = p \cdot (q \cdot r)$. In mathematischen Bereichen, in denen so etwas der Fall ist, wird daher auch manchmal von einem *Assoziator* gesprochen, weil es sich statt einer Aussage die gilt, eben um eine Operation handelt, die ein Datum produziert. Diese neue Vielfalt kann Probleme mit sich bringen und es ist daher üblich, sogenannte *Kohärenz* zu fordern. Dabei handelt es sich um natürlichen Gleichungen, die etwa zwischen verschiedenen Assoziatoren gelten sollten. Oder Gleichungen, die zwischen diesen Gleichheiten wieder gelten sollten. Falls Kohärenz gegeben ist, spricht man von *höheren Strukturen*, z.B. von höheren Monoiden, höheren Gruppen oder Ringen.

In der Homotopietypentheorie, die wir in der Vorlesung lernen, gibt es zwar keine bekannte Möglichkeit solche Kohärenz allgemein zu definieren, aber man kann zum Beispiel im Fall der Gleichheitstypen $x =_A y$ und den vorgestellten Operationen erfahrungsgemäß immer alle Kohärenzen konstruieren, die man gerade braucht. Ein Beispiel für eine Kohärenz, ein Level über der Assoziativität, ist das *MacLane Pentagon*:

Bemerkung 1.4.10
Seien A ein Typ, $x, y, z, w, u : A$. Dann gibt es für $p : x = y, q : y = z, r : z = w, s : w = u$ in $((p \cdot q) \cdot r) \cdot s = p \cdot (q \cdot (r \cdot s))$ zwei verschiedene natürliche Terme, zwischen denen sich eine Gleichheit konstruieren lässt.

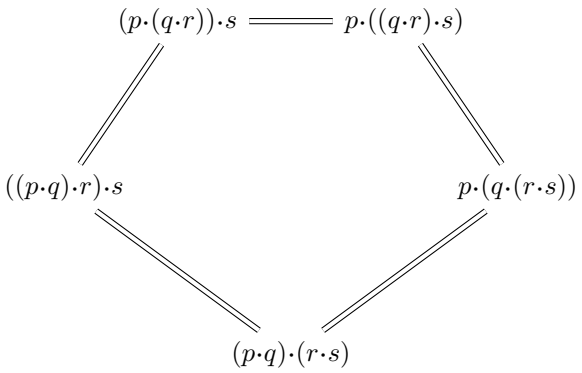


Abbildung 1: MacLane-Pentagon

Wir werden die Mittel erst noch einführen, mit denen diese Terme konstruiert werden können.

Wir wenden uns nun dem Verhalten von Gleichheit unter Abbildung mit Funktionen zu. Für jeden sinnvollen Begriff von Gleichheit, will man sicher, dass er von Funktionen respektiert wird. Für eine Funktion $f : A \to B$ und $p : x =_A y$ sollte auch $f(x) = f(y)$ gelten bzw. eine Gleichheit in $f(x) =_B f(y)$ kontruierbar sein. Das ist mit der bekannten Vorgehensweise schnell erledigt:

Definition 1.4.11
Seien A, B Typen, $f : A \to B$ eine Funktion, $x, y : A$ und $p : x =_A y$. Es sei $f(p) : f(x) =_B f(y)$ das *Bild der Gleichheit* p, das wir auch mit $\mathrm{ap}(f, p) : f(x) =_B f(y)$ bezeichnen, gegeben durch

$$f(\mathrm{refl}_x) :\equiv \mathrm{refl}_{f(x)}$$

Auch hierfür wollen wir zumindest die grundlegenden Gleichungen erwähnen:

Bemerkung 1.4.12
Seien A, B Typen und $f : A \to B$ eine Funktion.

(a) Für $x : A$ gilt $f(\mathrm{refl}_x) = \mathrm{refl}_{f(x)}$.

(b) Für $x, y : A$ und $p : x =_A y$ gilt $f(p^{-1}) = f(p)^{-1}$.

(c) Für $x, y, z : A$ und $p : x =_A y, q : y =_A z$ gilt $f(p \cdot q) = f(p) \cdot f(q)$.

Die Beweise folgen alle dem bekannten Schema.

Eine weitere entscheidende Eigenschaft, die bei Gleichheit erwarten würde, ist, dass sich gleiche Dinge auch in allen Eigenschaften gleichen. In leichter Abwandlung, könnte man auch sagen, wenn man eine Aussage $P(x)$ hat und zwei gleiche Objekte $x = y$, dann sollten $P(x)$ und $P(y)$ äquivalent sein. Dabei würde es wegen der Symmetrie auch schon reichen, zu verlangen, dass im Fall der Gleichheit $P(y)$ aus $P(x)$ folgt.

Für unsere Gleichheit bekommen wir für einen abhängigen Typen B sogar eine Abbildung $B(x) \to B(y)$ statt einer Implikation:

Definition 1.4.13
Seien A ein Typ und $x : A \vdash B(x)$ ein abhängiger Typ über A. Dann sei für $x, y : A$ und $p : x =_A y$ die Abbildung

$$\mathrm{tr}_B(p) : B(x) \to B(y)$$

gegeben durch $\mathrm{tr}(\mathrm{refl}_x) :\equiv \mathrm{id}_{B(x)}$. Die Abbildung $\mathrm{tr}_B(p)$ heißt *Transport* in B entlang von p.

Transport kann man für spezielle Typen konkretisieren.

Lemma 1.4.14
Seien A ein Typ und $a : A$. Für den von $x : A$ abhängigen Typ $B(x) :\equiv (x = a)$ kann der Transport für $x, x' : A$ und $p : x = x'$ berechnet werden:

$$((q : x = a) \mapsto p^{-1} \cdot q) = \mathrm{tr}_B(p) : B(x) \to B(x')$$

Beweis Die Behauptung folgt aus:

$$(q : x = a) \mapsto \mathrm{refl}_x^{-1} \cdot q = \mathrm{id}_{x=a}$$ □

Außerdem verträgt sich der Transport mit der Konkatenation:

Lemma 1.4.15
Seien A ein Typ und $x : A \vdash B(x)$. Für $x, y, z : A$ gilt:

$$\prod_{p:x=y} \prod_{q:y=z} \mathrm{tr}_B(q) \circ \mathrm{tr}_B(p) = \mathrm{tr}_B(p \cdot q)$$

Beweis Induktion über p. □

1.5 Abhängige Summen

Die abhängige Summe ist ein Typ, der die folgenden Konstruktionen und Aussagen in sich vereint:

$$\exists x \in M : P(x) \qquad A \times B \qquad \coprod_{i \in I} A_i$$

In der topologischen Anschauung entspricht die abhängige Summe dem Totalraum einer Faserung. Ähnlich wie $\prod_{x:A} B(x)$ den Funktionstyp $A \to B$ verallgemeinert, verallgemeinert die abhängige Summe das kartesische Produkt $A \times B$, dadurch, dass der rechte Faktor abhängig von $x : A$ variieren darf.

Regeln 1.5.1

- Für einen Typ A und $x : A \vdash B(x)$, kann die *abhängige Summe* $\sum_{x:A} B(x)$ gebildet werden. Die abhängige Summe wird auch *abhängiger Paartyp* oder *Sigmatyp* genannt und alternativ mit $(x : A) \times B(x)$ bezeichnet.
- Für $x : A$ und $b(x) : B(x)$ gibt es ein *Paar*, also ein Element $(x, b(x)) : \sum_{x:A} B(x)$.
- Für einen abhängigen Typ $y : \sum_{x:A} B(x) \vdash C(y)$ und einen abhängigen Term $x : A, b(x) : B \vdash c((x, b(x))) : C((x, b(x)))$ gibt es eine abhängige Funktion

$$\mathrm{ind}_{\sum}(C, c) : \prod_{y:\sum_{x:A} B(x)} C(y)$$

- Für auf diese Art definierte Funktionen gilt für $a : A, b : B(a)$ die Urteilsgleichheit

$$\mathrm{ind}_{\sum}(C, c)((a, b)) \equiv c((a, b)) : C((a, b))$$

Wenn keine Verwechslungsgefahr besteht, wollen wir etwa für "$c((a, b))$" auch einfach "$c(a, b)$" schreiben. Die Induktionsregel, also den dritten Punkt oben, hätten wir auch wie folgt formulieren können: Für $c : \prod_{x:A} \prod_{b:B(x)} C(a, b)$ gibt es $\mathrm{ind}_{\sum}(C, c) : \prod_{y:\sum_{x:A} B(x)} C(y)$. Und wir wollen auch hier wieder Funktionen durch ihre definierende Gleichung in folgender Form angeben:

$$g(a, b) :\equiv c(a, b)$$

Es gibt Projektionen auf die Faktoren einer abhängigen Summe:

Definition 1.5.2
Seien A ein Typ und $x : A \vdash B(x)$ abhängiger Typ über A.

(a) Die Funktion $\pi_1 : (\sum_{x:A} B(x)) \to A$ gegeben durch

$$\pi_1(a, b) :\equiv a$$

heißt *Projektion* auf die Basis oder Projektion auf den ersten Faktor.

(b) Die abhängige Funktion $\pi_2 : (y : \sum_{x:A} B(x)) \to B(\pi_1(y))$ gegeben durch

$$\pi_2(a, b) :\equiv b$$

heißt zweite *Projektion* oder Projektion auf den zweiten Faktor.

Wie bereits bei den Funktionen verwenden wir auch hier die übliche Notation, wenn keine echte Abhängigkeit vorliegt:

Definition 1.5.3
Für zwei Typen A und B schreiben wir auch $A \times B$ statt $\sum_{x:A} B(x)$ und sprechen vom *Produkt*[3] von A und B.

Mit Produkten lassen sich Konjuntionen ausdrücken. Ein Beispiel dafür ist die folgenden wichtigen Definitionen, die uns von nun an begleiten werden:

Definition 1.5.4
Seien A, B Typen.

(a) Zwei Funktionen $f : A \to B$ und $g : B \to A$ sind *zueinander invers*, wenn

$$\left(\prod_{x:A} g(f(x)) = x\right) \times \left(\prod_{y:B} f(g(y)) = y\right)$$

(b) Sei $f : A \to B$ eine Funktion. Wir sagen, dass f eine *Quasi-Inverse* oder *Inverse* hat, wenn der Typ

$$\mathrm{qinv(f)} :\equiv \sum_{g:B\to A} \left(\left(\prod_{x:A} g(f(x)) = x\right) \times \left(\prod_{y:B} f(g(y)) = y\right)\right)$$

einen Term hat.

[3] Die übliche Terminologie ist hier leider etwas verwirrend...

Mit Produkten können wir nun sogenanntes *Currying* formal fassen:

Definition 1.5.5
Seien A, B, C Typen. Die Abbildung

$$\text{curry} : ((A \times B) \to C) \to (A \to (B \to C))$$

ist gegeben durch:

$$\text{curry}(f) :\equiv (x \mapsto (y \mapsto f(x, y)))$$

Analog lässt sich auch *Uncurrying* definieren:

$$\begin{aligned}&\text{uncurry} : (A \to (B \to C)) \to ((A \times B) \to C)\\ &\text{uncurry}(f) :\equiv x \mapsto f(\pi_1(x), \pi_2(x))\end{aligned}$$

Bemerkung 1.5.6
Die Funktionen curry und uncurry sind zueinander invers. Allerdings können wir das und viele andere Aussagen dieser Form über Funktionen noch nicht mit der soweit eingeführten Typentheorie beweisen.

Mit abhängigen Summen haben wir die Möglichkeit, Existenzaussagen zu formulieren:

Definition 1.5.7
Seien $a, d : \mathbb{N}$. Die Phrase d *teilt* a steht für den folgenden Typen:

$$\sum_{c:\mathbb{N}} c \cdot d = a$$

Gleichungen in den natürlichen Zahlen werden wir uns später noch zuwenden. Zunächst wollen wir uns damit beschäftigen, wie sich für abhängige Summen oder speziell Produkte der Gleichheitstyp verhält.

Lemma 1.5.8
Seien A, B Typen.

(a) Für $x : A \times B$ gilt $x = (\pi_1(x), \pi_2(x))$.

(b) Seien $(a, b), (a', b') : A \times B$, dann gibt es zueinander inverse Funktionen:

$$\text{pair}_= : ((a =_A a') \times (b =_B b')) \to (a, b) =_{A\times B} (a', b')$$

und

$$\text{pair}_=^{-1} : (a, b) =_{A\times B} (a', b') \to ((a =_A a') \times (b =_B b'))$$

Beweis (a) Ziel ist es, einen Term in

$$\prod_{x:A\times B} x = (\pi_1(x), \pi_2(x))$$

zu konstruieren. Mit $\sum$-Induktion folgt dieser aus:

$$a \mapsto b \mapsto \text{refl}_{(a,b)} : \prod_{a:A}\prod_{b:B}(a, b) = (\pi_1(a, b), \pi_2(a, b))$$

Zur Verwendung in Teil (b), geben wir den damit konstruierten Term den Namen u.

(b) Wir definieren $\text{pair}_=$ durch doppelte Gleichheitsinduktion:

$$\text{pair}_=(\text{refl}_a, \text{refl}_b) :\equiv \text{refl}_{(a,b)}$$

und $\text{pair}_=^{-1}$ definieren wir zunächst etwas zu allgemein durch Bilder einer Gleichheit $p : x = y$ unter den Projektionen:

$$\text{pair}_=^{-1'}(p) :\equiv (\pi_1(p), \pi_2(p))$$

und setzen $\text{pair}_=^{-1}(p : (a, a') = (b, b')) :\equiv \text{pair}_=^{-1'}(p)$. Damit können wir mit Gleichheitsinduktion zeigen, dass $\text{pair}_=$ und $\text{pair}_=^{-1}$ zueinander invers sind. Ein Teil ist sofort erledigt:

$$\text{refl}_{(\text{refl}_a,\text{refl}_b)} : \text{pair}_=^{-1}(\text{pair}_=(\text{refl}_a, \text{refl}_b)) = (\text{refl}_a, \text{refl}_b)$$

Für den anderen bekommen konstruieren wir erstmal einen Term

$$t : \prod_{x,y:A\times B} \prod_{p:x=y} \mathrm{pair}_=(\mathrm{pair}_=^{-1'})(p) = u(\pi_1(x),\pi_2(x))^{-1}\cdot p\cdot u(\pi_1(y),\pi_2(y))$$

durch Gleichheits- und $\sum$-Induktion – wir dürfen also $\mathrm{refl}_{(a,b)}$ für p einsetzen:

$$\mathrm{pair}_=(\mathrm{pair}_=^{-1'})(\mathrm{refl}_{(a,b)}) = \mathrm{refl}_{(a,b)}^{-1}\cdot\mathrm{refl}_{(a,b)}\cdot\mathrm{refl}_{(a,b)}$$ □

Linke und rechte Seite sind per Definition gleich $\mathrm{refl}_{(a,b)}$, also müssen wir nur noch den somit konstruierten Term t entsprechend anpassen. Für $p : (a,b) = (a',b')$ lassen sich dazu folgende Gleichheiten konstruieren:

$$\begin{aligned}
\mathrm{pair}_=(\mathrm{pair}_=^{-1})(p) &\equiv \mathrm{pair}_=(\mathrm{pair}_=^{-1'})(p) && t_{(a,b),(a',b'),p}\\
&= u(a,b)^{-1}\cdot p\cdot u(a',b') && \text{Def.}\\
&\equiv \mathrm{refl}_{(a,b)}^{-1}\cdot p\cdot\mathrm{refl}_{(a',b')} && \text{Def.}\\
&\equiv p\cdot\mathrm{refl}_{(a',b')} && \text{Bemerkung 1.4.6}\\
&= p
\end{aligned}$$

Für allgemeine abhängige Summen gibt es eine ähnliche Möglichkeit, Gleichheiten zu erzeugen. Hier ist allerdings eine Schwierigkeit, dass zwei Elemente im zweiten Faktor, also etwa $b_x : B(x)$ und $b_y : B(y)$ im Allgemeinen nicht vom selben Typ sind. Also ergibt es keinen Sinn, nach einer Gleichheit $q : b_x = b_y$ zu fragen. Um b_x mit b_y zu vergleichen, können wir es entlang einer Gleichheit $p : x = y$ transportieren, also nach $\mathrm{tr}_B(p)(b_x) = b_y$ fragen.

Lemma 1.5.9
Seien A ein Typ und $x : A \vdash B(x)$ abhängig über A. Für $x, y : A$ und $b_x : B(x)$, $b'_y : B(y)$ gibt es eine Funktion

$$\sum_{=} : \prod_{p:x=y} \mathrm{tr}_B(p)(b_x) = b'_y \to (x,b_x) = (y,b'_y)$$

Beweis Mit Induktion über p reicht es, eine Funktion folgenden Typs anzugeben:

$$b_x = b'_x \to (x,b_x) = (x,b'_x)$$

Nun sind $b_x, b'_x : B(x)$ noch frei wählbar, also kann wieder Gleichheitsinduktion angewandt und der gesuchte Term auf $\mathrm{refl}_{(x,b_x)}$ festgelegt werden. □

Für die Gleichheit in abhängigen Summen gibt es auch eine Aussage wie Lemma 1.5.8, der wir uns jetzt aber erstmal nicht zuwenden.

1.6 Kontrahierbarkeit und Aussagen

Wir beschäftigen uns zunächst mit Gleichheit von Funktionen. In verschiedenen Zusammenhängen ist bereits die punkweise Gleichheit von Funktionen aufgetreten, der wir nun einen Namen geben wollen:

Definition 1.6.1
Seien A, B Typen und $f, g : A \to B$ Funktionen. Wir nennen f und g *homotop* oder *punktweise gleich*, wenn der folgenden Typ einen Term hat

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

Die Elemente von $f \sim g$ heißen *Homotopien* oder *punktweise Gleichheiten*.

Wir werden später das sogenannte *Univalenzaxiom* und ein *Intervall* als höheren induktiven Typen einführen. Aus beidem kann jeweils gefolgert werden, dass punktweise Gleichheiten bereits die Gleichheit von Funktionen zur Folge hat. Letztere Aussage hat einen eigenen Namen und wird von uns im Folgenden als Axiom verwendet:

Axiom 1.6.2 (Funktionsextensionalität)
Unter *Funktionsextensionalität* versteht man einen Term, der für Typen A, B und Funktionen $f, g : A \to B$ wie folgt gegeben ist:

$$\mathrm{FunExt}_{f,g} : \left(\prod_{x:A} f(x) = g(x)\right) \to f = g$$

(Genauer fordern wir, dass $\mathrm{FunExt}_{f,g}$ und $(p : f = g) \mapsto ((x : A) \mapsto \mathrm{ap}(h \mapsto h(x), p)) : (f = g) \to f \sim g$ invers zueinander sind, aber das brauchen wir nicht, bis wir FunExt sowieso nochmal auf eine andere Art einführen.) Wir werden uns jeweils merken, für welche Aussagen wir dieses Axiom brauchen.

Bemerkung 1.6.3
Mit FunExt sind die Funktionen curry und uncurry aus Definition 1.5.5 zueinander invers.

Was wir bereits über Gleichheit wissen und mit Gleichheiten machen können, lässt sich leicht auf Homotopien jeweils punktweise übertragen:

Bemerkung 1.6.4
Seien A, B Typen und $f, g : A \to B$ Funktionen.

(a) Es gibt stets den folgenden Term:

$$(x : A) \mapsto \mathrm{refl}_{f(x)} : f \sim f$$

(b) Es gibt eine Operation

$$H \mapsto (x \mapsto H(x))^{-1}) : f \sim g \to g \sim f$$

(c) Für eine weitere Funktion $h : A \to B$ gibt es eine Operation

$$H \mapsto H' \mapsto (x \mapsto H(x) \cdot H'(x)) : f \sim g \to g \sim h \to f \sim h$$

Ziel dieses Abschnitts ist es, die ersten beiden Stufen einer Hierarchie auf allen Typen einzuführen und zu untersuchen. Diese Hierarchie der sogenannten *n-Typen* ist nicht total und bezieht sich auf die Komplexität der Gleichheiten in Typen. In der niedrigsten Stufe sind alle Elemente eines Typs gleich einem fest gewählten, wir werden sehen, dass eine Konsequenz davon ist, dass auch Gleichheiten zwischen Gleichheiten keine neue Komplexität aufweisen. Wir nennen diese einfachsten Typen kontrahierbar und wollen gleich die nächsten beiden Stufen benennen:

Definition 1.6.5
Sei A ein Typ.

(a) A heißt *kontrahierbar* oder -2-*Typ*, wenn

$$\mathrm{isContr}(A) :\equiv \sum_{c:A} \prod_{x:A} x = c$$

(einen Term hat).

(b) A heißt *Aussage* oder -1-*Typ*, wenn

$$\mathrm{isProp}(A) :\equiv \prod_{x,y:A} x = y$$

(c) A heißt 0-*Typ* oder *Menge*, wenn

$$\mathrm{isSet}(A) :\equiv \prod_{x,y:A} \prod_{p,q:x=y} p = q$$

Beispiel 1.6.6

(a) $\mathbf{1}$ ist kontrahierbar.

(b) $\emptyset$ ist eine Aussage.

Für die kontrahierbaren Typen stellt sich zusätzlich zum bereits Gesagten heraus, dass jeder kontrahierbare Typ bereits mehr oder weniger der Einheitstyp ist:

Bemerkung 1.6.7
Sei A ein kontrahierbarer Typ. Es gilt:

(a) Für $x, y : A$ ist $x =_A y$ kontrahierbar.

(b) Es gibt Funktionen $f : A \to \mathbf{1}$ und $g : \mathbf{1} \to A$, die zueinander invers sind.

(c) A ist eine Aussage.

Beweis (a), (b) In den Übungen.
(c) Wir haben

$$k : \mathrm{isContr}(A) \equiv \sum_{c:A} \prod_{x:A} x = c$$

und damit:

$$x \mapsto y \mapsto k_x \cdot k_y^{-1} : \prod_{x,y:A} x = y$$

□

Mit der folgenden Bemerkung scheinen auch Aussagen wenig Vielfalt zuzulassen:

Bemerkung 1.6.8
Sei P eine Aussage und $t : P$. Dann ist P kontrahierbar.

Beweis Es gibt

$$a : \prod_{x,y:P} x = y$$

und damit

$$(t, x \mapsto a_{x,t}) : \mathrm{isContr}(P) \equiv \sum_{c:P} \prod_{x:P} x = c$$

□

Es gibt einen überraschenderweise kontrahierbaren Typen, der uns noch begleiten wird.

Lemma 1.6.9
Seien A ein Typ. Dann ist für $y : A$ der Typ

$$\sum_{x:A} x = y$$

kontrahierbar.

Beweis Wegen $(y, \mathrm{refl}_y) : \sum_{x:A} x = y$ reicht es zu zeigen, dass der Typ eine Aussage ist, also je zwei Elemente gleich sind. Durch $\sum$-Induktion, reicht es zu zeigen, dass je zwei Paare

$$(x, q), (x', q') : \sum_{x:A} x = y$$

gleich sind. Es ist $q \cdot q'^{-1} : x = x'$ und damit können wir $\sum_{=}$ verwenden:

$$\Sigma_{=} : \prod_{p:x=x'} \mathrm{tr}_{x:A \vdash x=y}(p)(q) = q' \to (x, q) = (x', q')$$

Den Transport haben wir in Lemma 1.4.14 berechnet und können einsetzen:

$$(q \cdot q'^{-1})^{-1} \cdot q = q' \to (x, q) = (x', q')$$

Mit Induktion kann berechnet werden: $(q \cdot q'^{-1})^{-1} = q' \cdot q^{-1}$ und damit mit den Rechengesetzen für Gleichheiten der Beweis beendet werden. □

Es gibt einige Konstruktionen, die die Eigenschaften "Aussage" erhalten (allgemeiner auch für "n-Typ").

Lemma 1.6.10
Seien P, Q Aussagen und A ein beliebiger Typ.

(a) Mit FunExt gilt: Der Typ $A \to P$ ist eine Aussage.

(b) $P \times Q$ ist eine Aussage.

Beweis (a) Sei $t : \mathrm{isProp}(P)$, dann ist

$$\mathrm{FunExt}_{f,g}((x : A) \mapsto t_{(f(x),g(x))}) : \mathrm{isProp}(A \to P) \equiv \prod_{f,g:A\to P} f = g$$

(b) Seien $t : \mathrm{isProp}(P)$ und $s : \mathrm{isProp}(Q)$. Per doppelter $\sum$-Induktion reicht es für $p, p' : P$ und $q, q' : Q$ zu zeigen:

$$(p, q) = (p', q')$$

was durch $\mathrm{pair}_=(t_{p,p'}, s_{q,q'})$ gegeben ist. □

Es ist möglich, Typen universell in eine Aussage zu verwandeln. Dafür werden wir zunächst nur eine Formierungs- und Einführungsregel kennenlernen:

Regeln 1.6.11 (-1-Truncation, teilweise)
Für jeden Typ A gibt es eine Aussage $\|A\|$. Weiter gibt es für jedes $a : A$ ein $|a| : \|A\|$. Der Typ $\|A\|$ heißt *-1-Abschneidung* oder *-1-Truncation*. Mit den weiteren Regeln wird sich rausstellen, dass $|\_| : A \to \|A\|$ genau dann eine Inverse hat, wenn A bereits eine Aussage war.

Damit können wir alle üblichen Logikkonstrukte für Aussagen definieren:

Definition 1.6.12
Seien P und Q Aussagen.
(a) $P \Rightarrow Q :\equiv P \to Q$

(b) $P \wedge Q :\equiv P \times Q$

(c) $P \vee Q :\equiv \|P \sqcup Q\|$

(d) $\neg P :\equiv P \to \emptyset$

Sei nun A ein Typ und $P(x)$ eine Aussage für $x : A$.
(a) $\forall x : A$ gilt $P(x) :\equiv \prod_{x:A} P(x)$

(b) $\exists x : A$ mit $P(x) :\equiv \|\sum_{x:A} P(x)\|$

Bei dieser Gelegenheit sollte festgehalten werden, dass es möglich ist, nicht konstruktive Axiome anzunehmen, um unsere Typentheorie bei Bedarf zu spezialisieren. Um diese zu formulieren brauchen wir aber noch manche der folgenden, sowieso wichtigen, Begriffe:

Definition 1.6.13
Seien A, B Typen und $f : A \to B$ eine Funktion.
(a) Für $b : B$ ist die *Faser* von f über b gegeben durch:

$$\mathrm{fib}_f(b) :\equiv f^{-1}(b) :\equiv \sum_{x:A} f(x) = b$$

(b) f heißt *injektiv*, wenn

$$\prod_{y:B} \mathrm{isProp}(f^{-1}(y))$$

(c) f heißt *surjektiv*, wenn

$$\prod_{y:B} \|f^{-1}(y)\|$$

(d) f heißt *Äquivalenz*, wenn

$$\mathrm{isEquiv}(f) :\equiv \prod_{y:B} \mathrm{isContr}(f^{-1}(y))$$

Bemerkung 1.6.14
Es ist möglich, bei Bedarf anzunehmen, dass die folgenden klassischen Axiome gelten
(a) Das Gesetz vom *ausgeschlossenen Dritten*: Für jede Aussage gilt $P \vee \neg P$

(b) Das Auswahlaxiom: Für jede surjektive Funktion $f : A \to B$ zwischen Mengen A und B gibt es $s : B \to A$ mit $f \circ s \sim \mathrm{id}_B$.

Wir werden noch sehen, dass es eine gute Idee ist, das Auswahlaxiom auf Abbildungen zwischen Mengen zu beschränken. Jetzt wollen wir noch den Abschnitt mit zwei Bemerkungen zur Äquivalenz beenden:

Bemerkung 1.6.15
Seien A, B Typen und $f : A \to B$. Wenn f surjektiv und injektiv ist, dann ist f eine Äquivalenz.

In Abschnitt 2.1 werden wir beweisen, dass jede Funktion mit einer beidseitigen Inversen auch eine Äquivalenz ist.

1.7 Universen

Ein *Universum* kann man sich erstmal als Typ aller Typen vorstellen. Problematisch ist dabei, wie in der Mengenlehre, dass das zu Widersprüchen führt. Eine einfach Lösung, die wir hier verwenden werden ist die leichte Abwandlung zu sagen, dass jeder Typ in *einem* Universum liegt und Universen unter möglichst vielen Konstruktionen abgeschlossen sind. Genauer fordern wir eine abzählbare Hierarchie von Universen:

$$\mathcal{U}_0, \mathcal{U}_1, \dots$$

Dabei sind die Indizes allerdings nicht unsere natürlichen Zahlen, sondern sogenannte *Universenlevel*. Dabei handelt es sich um eine Variante der natürlichen Zahlen, die implizit durch Regeln gegeben ist. Das verhindert zum Beispiel die Konstruktion von aufsteigenden Folgen von Universen.

Da es mit Universen möglich ist, auszudrücken, dass etwas ein Typ ist, können wir von nun auf dieses Urteil verzichten. Wir verwenden also in Zukunft "$A : \mathcal{U}_i$" synonym mit "A ist ein Typ".

Regeln 1.7.1
Wir können nun in allen Regeln Urteile der "A Typ" ersetzen durch "$A : \mathcal{U}_i$". Bei der Regel ΠF bedeutet das etwa:

$$\frac{\Gamma \vdash A : \mathcal{U}_i \qquad \Gamma, x : A \vdash B(x) : \mathcal{U}_i}{\Gamma \vdash \prod_{x:A} B(x) : \mathcal{U}_i} \quad (\Pi\text{F})$$

Dabei haben wir die Regel etwas ausgebaut, um fordern zu können, dass A und die $B(x)$ im gleichen Universum liegen. Der Index i ist dabei 0 oder $j + 1$ für einen erlaubten Index j. Die Universen gibt es jetzt auch einfach so als Typen:

$$\frac{\Gamma \text{ Kontext}}{\Gamma \vdash \mathcal{U}_i : \mathcal{U}_{i+1}} \quad (\mathcal{U}\text{F})$$

Insbesondere können wir nun abhängige Typen auch immer äquivalent als Funktionen in ein Universum ausdrücken:

$$\frac{\Gamma \vdash A : \mathcal{U}_i \qquad \Gamma, x : A \vdash B(x) : U_i}{\Gamma \vdash x \mapsto B(x) : A \to \mathcal{U}_i}$$

Grundsätzlich kann es vorkommen, dass wir auch mal Typen aus Bestandteilen konstruieren wollen, die in unterschiedlichen Universen liegen. Dazu können wir mit der folgenden Regel alles in ein passendes Universum schieben:

$$\frac{\Gamma \vdash A : \mathcal{U}_i}{\Gamma \vdash A : \mathcal{U}_{i+1}}$$

Den Rest dieses Abschnitts verbringen wir damit, Neuerungen zu erwähnen, die sich durch die Verwendung von Universen ergeben. Interessanterweise haben wir erst durch Universen die Möglichkeit, manche Gleichheitstypen zu charakterisieren, was wir aber erst im nächsten Abschnitt ausführlich angehen werden.

Bemerkung 1.7.2
Mit Universen können wir Induktion als abhängige Funktion umschreiben, etwa für $\mathbb{N}$:

$$\mathrm{ind}_{\mathbb{N}} : \prod_{P:\mathbb{N}\to\mathcal{U}_i} P(0_{\mathbb{N}}) \to \Big(\prod_{n:\mathbb{N}} P(n) \to P\big(\mathrm{succ}_{\mathbb{N}}(n)\big)\Big) \to \Big(\prod_{n:\mathbb{N}} P(n)\Big)$$

Konvention 1.7.3
Meistens werden wir den Index der Universen weglassen, also etwa nur $A : \mathcal{U}$ schreiben. Situationen in denen die Universenlevel wichtig sind, werden eher selten sein, kommen aber vor.

Beispiel 1.7.4
Dank Universen können wir nun abhängige Typen über Rekursion definieren:

$$\begin{aligned} B &: \mathbf{2} \to \mathcal{U} \\ B(0_{\mathbf{2}}) &:\equiv \emptyset \\ B(1_{\mathbf{2}}) &:\equiv \mathbf{1} \end{aligned}$$

2 Univalenz

2.1 Äquivalenzen

In diesem Abschnitt werden wir drei verschiedene Definitionen von Äquivalenzen einführen und zeigen, dass es sich um den selben Begriff handelt. Einen haben wir bereits eingeführt - Funktionen mit kontrahierbaren Fasern heißen Äquivalenz. Von einem Äquivalenzbegriff erwarten wir, dass es sich beim entsprechenden Typ $\mathrm{isEquiv}(f)$ um eine Aussage handelt. Das hat zur Folge, dass wir es als eine Eigenschaft von Funktionen anssehen können eine Äquivalenz zu sein. Damit ist dann auch der Typ der Äquivalenzen zwischen A und B eine Untertyp von $A \to B$, d.h. die Vergiß-Abbildung

$$\pi_1 : \left(\sum_{f:A\to B} \mathrm{isEquiv}(f) \right) \to (A \to B)$$

ist injektiv.

Zur Vorbereitung beschäftigen wir uns noch etwas mit Homotopien.

Bemerkung 2.1.1
Seien $A, B : \mathcal{U}$ und $f, g : A \to B$.

(a) Sei $H : f \sim g$. Für $\varphi : A' \to A$ gibt es eine Homotopie $H_{\varphi(\_)} : f \circ \varphi \sim g \circ \varphi$ und für $\psi : B \to B'$ gibt es eine Homotopie $\psi(H) : \psi \circ f \sim \psi \circ g$. Diese Operationen nennt man *whiskering*.

(b) Sei $H : f \sim g$. Homotopien sind natürlich in folgendem Sinn: Für $p : x =_A y$ gilt:

$$H_x \cdot g(p) = f(p) \cdot H_y$$

(c) Für eine Homotopie $H : f \sim \mathrm{id}$ gilt $f(H_x) = H_{f(x)}$.

Wir beginnen mit einer scheinbar unbedeutenden Variation des Begriffs Quasi-Inverse und legen dabei mehr Notation für Äquivalenzen fest:

Definition 2.1.2
Seien $A, B : \mathcal{U}$ und $f : A \to B$.

(a) f hat eine *Linksinverse*, wenn es ein $g : B \to A$ gibt und $g \circ f \sim \mathrm{id}_A$.

(b) f hat eine *Rechtsinverse* oder einen *Schnitt*, wenn es ein $h : B \to A$ gibt und $f \circ h \sim \mathrm{id}_B$.

(c) f ist eine *Äquivalenz* oder eine Funktion mit *Links- und Rechtsinversen*, wenn

$$\mathrm{isEquiv}(f) :\equiv \mathrm{LRInv}(f) :\equiv \left(\sum_{g:B\to A} g \circ f \sim \mathrm{id}_A \right) \times \left(\sum_{h:B\to A} f \circ h \sim \mathrm{id}_B \right)$$

(d) Wenn es eine Äquivanz zwischen A und B gibt, dann sagen wir A ist *äquivalent* zu B.

(e) Für den Typ der Äquivalenzen zwischen A und B schreiben wir:

$$A \simeq B :\equiv \sum_{f:A\to B} \mathrm{isEquiv}(f)$$

Es wird sich herausstellen, dass $\mathrm{LRInv}(f)$ und $\mathrm{qinv}(f)$ für manche Funktionen keine äquivalenten Typen sind. Insbesondere werden wir noch sehen, dass immer $\mathrm{LRInv}(f)$ eine Aussage ist, $\mathrm{qinv}(f)$ aber keine Aussage mehr sein muss, wenn A und B komplizierte Gleichheitstypen haben. Die beiden Begriffe sind aber logisch äquivalent, d.h. es gibt für jedes $f : A \to B$ Abbildungen $\mathrm{LRInv}(f) \to \mathrm{qinv}(f)$ und $\mathrm{qinv}(f) \to \mathrm{LRInv}(f)$. Zunächst halten wir die Vokabel "logisch äquivalent" fest:

Definition 2.1.3
Zwei Typen A und B heißen *logisch äquivalent*, wenn es Funktionen $f : A \to B$ und $g : B \to A$ gibt.

Bei Aussagen reicht die logische Äquivalenz bereits, damit die Typen äquivalent sind. Damit werden die verschiedenen Varianten des Typs "isEquiv(f)", die wir in diesem Abschnitt kennenlernen, äquivalente Typen sein, wenn wir logische Äquivalenz gezeigt haben und dass es sich jeweils um Aussagen handelt. Letzteres werden wir allerdings noch aufschieben.

Bemerkung 2.1.4
Seien $A, B : \mathcal{U}$ und $f : A \to B$. Die Typen $\mathrm{LRInv}(f)$ und $\mathrm{qinv}(f)$ sind logisch äquivalent.

Beweis Nehmen wir zünachst an, es gibt eine Quasi-Inverse $g : B \to A$ mit $H : g \circ f \sim \mathrm{id}_A$ und $K : f \circ g \sim \mathrm{id}_B$. Dann ist

$$((g, H), (g, K)) : \mathrm{LRInv}(f)$$

Seien nun andererseits $g, h : B \to A$ links- und rechtsinvers zu f. Wir zeigen, dass g und h homotop sind. Damit können wir dann zeigen, dass g auch rechtsinvers ist. Durch Whiskering erhalten wir:

$$g \sim g \circ (f \circ h) \sim (g \circ f) \circ h \sim h$$

Also ist g rechtsinvers:

$$f \circ g \sim f \circ h \sim \mathrm{id}_B$$ □

Wenn wir also zu einer Funktion $f : A \to B$ eine beidseitige Inverse konstruieren, dann ist f eine Äquivalenz (im Sinn einer Funktion mit Links- und Rechtsinversen). Äquivalenzen haben die Operationen, die wir bereits von Gleichheiten kennen:

Bemerkung 2.1.5
(a) Die Identität ist eine Äquivalenz.

(b) Eine Inverse einer Äquivalenz ist eine Äquivalenz.

(c) Seien $A, B, C : \mathcal{U}$. Wenn $f : A \to B$ und $g : B \to C$ Äquivalenzen sind, dann ist $g \circ f : A \to C$ eine Äquivalenz.

Beweis Wir zeigen die Aussagen für den Äquivalenzbegriff der Funktion mit Links- und Rechtsinversen. Mit Theorem 2.1.12 gilt damit das gleiche für die anderen Äquivalenzbegriffe.
(a) Die Identität ist ihre eigene Links- und Rechtsinverse. Die nötigen Homotopien gelten schon als urteilsmäßige von Funktionen.

(b) Wenn $f : A \to B$ eine Linksinverse $g : B \to A$ und eine Rechtsinverse $h : B \to A$ hat, dann ist bekommen wir mit Bemerkung 2.1.4 eine Quasi-Inverse $f^{-1} : B \to A$. Und f^{-1} ist eine Äquivalenz mit Linksinverser f und Rechtsinverser f.

(c) Auf Übungsblatt 4 wird das für Quasi-Inversen gezeigt. Zusammen mit Bemerkung 2.1.4 gilt das also auch für Äquivalenzen. □

Beispiel 2.1.6 (Transport ist Äquivalenz)
Für $A : \mathcal{U}$, $B : A \to \mathcal{U}$, $x, y : A$ und jedes $p : x = y$ ist $\mathrm{tr}_B(p) : B(x) \to B(y)$ eine Äquivalenz mit Inverser $\mathrm{tr}_B(p^{-1}) : B(y) \to B(x)$.

Wir wollen nun den Äquivalenzbegriff der Links- und Rechtsinvertierbarkeit mit dem der kontrahierbaren Fasern in Zusammenhang bringen. Wir werden allerdings erstmal eine Implikation klären und für die zweite einen dritten Äquivalenzbegriff einführen.

Bemerkung 2.1.7
Seien $A, B : \mathcal{U}$ und $f : A \to B$. Wenn alle Fasern von f kontrahierbar sind, dann hat f Links- und Rechtsinverse, es gibt also eine Funktion:

$$\left(\prod_{y:B} \mathrm{isContr}(f^{-1}(y)) \right) \to \mathrm{LRInv}(f)$$

Beweis Sei $k : \prod_{y:B} \text{isContr}(f^{-1}(y))$. Wir konstruieren eine beidseitige Inverse $g : B \to A$ indem wir $y : B$ auf das Kontraktionszentrum der Faser $f^{-1}(y)$ abbilden: Für $\pi_1(k_y)$ ist ein Element von $f^{-1}(y)$, besteht also aus $x : A$ und $p : f(x) = y$, also können wir festlegen

$$g(y) :\equiv x$$

Dann ist $f(g(y)) = f(x) = y$, also g Rechtsinverse von f. Für $x : A$ sei $x' :\equiv g(f(x))$. Es müssen $x : A$ und $x' : A$ beide in $f^{-1}(f(x))$ liegen und da diese Faser kontrahierbar ist, gibt es eine Gleichheit $q : x = x' = g(f(x))$. Genauer haben wir durch die Kontrahierbarkeit eine Gleichheit $q' : (x, \text{refl}_{f(x)}) = (x', p')$ in der Faser $f^{-1}(f(x))$. Diese Gleichheit können wir mit $\pi_1 : f^{-1}(f(x)) \to A$ abbilden, also $q :\equiv \pi_1(q')$ setzen. □

Die Umkehrung dieser Aussage ist komplizierter zu zeigen und wir werden zunächst zeigen, dass sich Links- Rechtsinverse in eine sogenannte kohärente Inverse überstzen lassen. Kohärente Inverse $g : B \to A$ einer Funktion $f : A \to B$ haben eine Kompatiblität zwischen den beiden Homotopien $f \circ g \sim \text{id}$ und $g \circ f \sim \text{id}$. Für eine Funktion $f : A \to B$ mit Inverser $g : B \to A$ besagt diese Kohärenz, dass die beiden Möglichkeiten eine Homotopie $f \circ g \circ f \sim f$ zu konstruieren punktweise gleich sind.

Definition 2.1.8
Seien $A, B : \mathcal{U}$ und $f : A \to B$. Eine Funktion $g : B \to A$ heißt *kohärente Inverse* von f, wenn es Homotopien $H : g \circ f \sim \text{id}$ und $K : f \circ g \sim \text{id}$ gibt und

$$\text{koh} : \prod_{x:A} f(H_x) = K_{f(x)}$$

Wir schreiben $\text{CohInv}(f)$ für den Typ der *kohärenten Inversen* von f.

Wir wollen direkt einsehen, dass wir von dieser Sorte Äquivalenz die Kontrahierbarkeit der Fasern beweisen können:

Bemerkung 2.1.9
Seien $A, B : \mathcal{U}$ und $f : A \to B$. Wenn f eine kohärente Inverse hat, dann sind die Fasern von f kontrahierbar.

Vor dem Beweis noch ein Hilfslemma, das wir auch sonst noch wiederverwenden können:

Lemma 2.1.10
Seien A, B Typen, $f : A \to B$ und $y : B$. Für $(x, q), (x', q') : f^{-1}(y)$ gibt es eine Funktion:

$$\prod_{p:x =_A x'} f(p)^{-1} \cdot q = q' \to (x, q) = (x', q')$$

Beweis (von Lemma 2.1.10) Induktion über p und $\sum_{=}$. □

Beweis (von 2.1.9) Habe also $f : A \to B$ eine kohärente Inverse, gebe es also $g : B \to A$, $H : g \circ f \sim \text{id}$, $K : f \circ g \sim \text{id}$ und $\text{koh} : \prod_{x:A} f(H_x) = K_{f(x)}$. Wir müssen nun für jede Faser von f eine Kontraktion angeben. Als Kontraktionszentrum für die Faser über $y : B$ wählen wir

$$k_y :\equiv (g(y), K_y) : f^{-1}(y)$$

Seien nun also $(x, q), (x', q') : f^{-1}(y)$. Damit haben wir auch $q \cdot q'^{-1} : f(x) = f(x')$ und damit:

$$H_x^{-1} \cdot g(q \cdot q'^{-1}) \cdot H'_x : x = x'$$

Darauf wenden wir jetzt f an:

$$f(H_x^{-1} \cdot g(q \cdot q'^{-1}) \cdot H'_x) : f(x) = f(x')$$

und berechnen mit Natürlichkeit von Homotopien, der Kohärenz und den Gruppoidgesetzen:

$$\begin{aligned} f(H_x^{-1} \cdot g(q \cdot q'^{-1}) \cdot H'_x) &= f(H_x^{-1}) \cdot f(g(q \cdot q'^{-1})) \cdot f(H'_x) \\ &= f(H_x^{-1}) \cdot K_{f(x)} \cdot q \cdot q'^{-1} \cdot K^{-1}_{f(x')} \cdot f(H'_x) \\ &= q \cdot q'^{-1} \end{aligned}$$

Mit dem Lemma haben wir also die gewünschte Gleichheit. □

Bemerkung 2.1.11
Seien $A, B : \mathcal{U}$ und $f : A \to B$ habe eine beidseitige Inverse $g : B \to A$. Dann ist g auch eine kohärente Inverse von f.

Beweis Seien $g : B \to A$ und $H : g \circ f \sim \mathrm{id}$, $K : f \circ g \sim \mathrm{id}$. Setze für $y : B$:

$$K'(y) :\equiv K^{-1}_{f(g(y))} \cdot f(H_{g(y)}) \cdot K_y$$

und rechne nach dass für $x : A$ gilt: $f(H_x) = K'_{f(x)}$. □

Theorem 2.1.12
Seien $A, B : \mathcal{U}$ und $f : A \to B$, dann sind die folgenden Äquivalenzbegriffe logisch äquivalent:
(a) Alle Fasern von f sind kontrahierbar:

$$\mathrm{isEquiv}(f) :\equiv \prod_{y:B} \mathrm{isContr}(f^{-1}(y))$$

(b) f hat eine Linksinverse und eine Rechtsinverse:

$$\mathrm{isEquiv}(f) :\equiv \mathrm{LRInv} \equiv \left(\sum_{g:B\to A} g \circ f \sim \mathrm{id}_A \right) \times \left(\sum_{h:B\to A} f \circ h \sim \mathrm{id}_B \right)$$

(c) f hat eine kohärente Inverse:

$$\mathrm{isEquiv}(f) :\equiv \mathrm{CohInv}(f) :\equiv \sum_{g:B\to A} \sum_{H:g\circ f\sim \mathrm{id}} \sum_{K:f\circ g\sim \mathrm{id}} \prod_{x:A} f(H_x) = K_{f(x)}$$

Wir schreiben für alle drei Begriffe $\mathrm{isEquiv}(f)$, weil sich noch rausstellen wird, dass alle diese Typen Aussagen sind und damit alle drei Typen äquivalent sind.

Beweis Die Bemerkungen dieses Abschnitts beweisen per Ringschluss die logische Äquivalenz. □

2.2 Univalenz

In diesem Abschnitt werden wir uns mit den Universen und ihren Gleichheitstypen beschäftigen. Eine historisch wichtige Idee von Vladimir Voevodsky ist das *Univalenzaxiom*, das die Gleichheitstypen im Universum mit den Äquivalenzen von Typen identifiziert. Dieses Axiom werden wir kennenlernen und Konsequenzen daraus ziehen. Davor wollen wir aber noch allgemeine Konsequenzen aus der Existenz von Universen ziehen.

Mit Universen sind wir in der Lage Ungleichheit von Elementen eines Typs zu beweisen. Diese definieren wir zunächst als Negation der Gleichheit:

Definition 2.2.1
Seien $A : \mathcal{U}$ und $x, y : A$. Dann ist x *ungleich* y, wenn

$$x \neq y :\equiv (x = y \to \emptyset).$$

Man beachte, dass es sich bei diesem Typ stets um eine Aussage handelt. Es sind also in $x \neq y$ keine so interessanten Dinge zu finden, wie im Gleichheitstyp und wir werden uns auch wenig mit Ungleichheiten beschäftigen.

Bemerkung 2.2.2
Es gilt $1_{\mathbf{2}} \neq 0_{\mathbf{2}}$.

Beweis Per Rekursion können wir uns den folgenden abhängigen Typen definieren:

$$B : \mathbf{2} \to \mathcal{U}$$
$$B(0_{\mathbf{2}}) :\equiv \emptyset$$
$$B(1_{\mathbf{2}}) :\equiv \mathbf{1}$$

Um die Ungleichheit zu zeigen, dürfen wir $p : 1_{\mathbf{2}} = 0_{\mathbf{2}}$ annehmen und müssen ein Element in $\emptyset$ konstruieren. Sei also $p : 1_{\mathbf{2}} = 0_{\mathbf{2}}$. Damit haben wir auch eine Abbildung:

$$\mathrm{tr}_B(p) : B(1_{\mathbf{2}}) \to B(0_{\mathbf{2}})$$

Und wegen $* : B(1_{\mathbf{2}}) \equiv \mathbf{1}$ haben wir auch ein Element, das wir in diese Abbildung einsetzen können. Es ist also $\mathrm{tr}_B(p)(*) : B(0_{\mathbf{2}}) \equiv \emptyset$. □

Dieser Trick lässt sich auf alle Elemente induktiver Typen übertragen, für die wir "verschiedene" Werte vorgeben können. Für einzelne natürliche Zahlen etwa:

Bemerkung 2.2.3
Es gilt $0_{\mathbb{N}} \neq 1_{\mathbb{N}} :\equiv \mathrm{succ}_{\mathbb{N}}(0_{\mathbb{N}})$.

Beweis Sei

$$\begin{aligned} B &: \mathbb{N} \to \mathcal{U} \\ B(0_{\mathbb{N}}) &:\equiv \emptyset \\ B(1_{\mathbb{N}}) &:\equiv \mathbf{1} \end{aligned}$$

womit für $p : 0_{\mathbb{N}} = 1_{\mathbb{N}}$ ein Element $\mathrm{tr}_B(p^{-1})(*) : \emptyset$ gegeben ist. □

Typischerweise interessiert man sich für eine vollständige Charakterisierung der Gleichheitstypen eines Typs. Für die natürlichen Zahlen wäre das ein in $n, k : \mathbb{N}$ abhängiger Typ $B(n, k)$ sodass $B(n, k) \simeq (n =_{\mathbb{N}} k)$. Das wird unser Ziel in Abschnitt 2.4 sein.

Ohne Weiteres können wir bereits für $A, B : \mathcal{U}_i$ den Typ $A =_{\mathcal{U}_i} B$ der Gleichheiten zwischen A und B formen. Dieser liegt allerdings im nächsthöheren Universum $\mathcal{U}_{i+1}$, da wir in der Formierungsregel für Gleichheitstypen "X ist ein Typ" durch "$X : \mathcal{U}_i$" ersetzen und dafür nur "$\mathcal{U}_i : \mathcal{U}_{i+1}$" in Frage kommt:

$$\frac{\Gamma \vdash \mathcal{U}_i : \mathcal{U}_{i+1} \qquad \Gamma \vdash A, B : \mathcal{U}_i}{\Gamma \vdash A =_{\mathcal{U}_i} B : \mathcal{U}_{i+1}} \quad (=\mathrm{F})$$

Damit lässt sich zeigen, dass gleiche Typen äquivalent sind.

Bemerkung 2.2.4
Für $A, B : \mathcal{U}$ gibt es eine Funktion

$$\mathrm{tr}_{\mathrm{id}_{\mathcal{U}}} : A =_{\mathcal{U}} B \to (A \simeq B)$$

den *Universentransport*.

Beweis Seien $A, B : \mathcal{U}_i$. Dann gibt es einen abhängigen Typ $X : \mathcal{U}_i \to \mathcal{U}_{i+1}$ und damit

$$\mathrm{tr}_{\mathrm{id}_{\mathcal{U}}} :\equiv \mathrm{tr}_X : A =_{\mathcal{U}_i} B \to A \simeq B$$

Nach Beispiel 2.1.6 sind Transporte immer Äquivalenzen. □

Es ist also möglich, aus einer Gleichheit von Typen eine Äquivalenz zu kontruieren. Die Umkehrung, aus einer Äquivalenz auch eine Gleichheit konstruieren zu können, nennt man das Univalenzaxiom.

Axiom 2.2.5 (Univalenzaxiom)
Von nun an nehmen wir an, dass das *Univalenzaxiom* gilt: Für $A, B : \mathcal{U}$ ist die Funktion

$$\mathrm{tr}_{\mathrm{id}_{\mathcal{U}}} : A =_{\mathcal{U}} B \to A \simeq B$$

eine Äquivalenz ist. Die Inverse von $\mathrm{tr}_{\mathrm{id}_{\mathcal{U}}}$ bezeichnen wir mit ua:

$$\mathrm{ua} : A \simeq B \to A =_{\mathcal{U}} B$$

Wir nehmen an, dass das Univalenzaxiom für jedes Universum gilt.

Mit Univalenz ergibt sich, dass es tatsächlich verschiedene Gleichheiten zwischen Elementen geben kann. Um dafür ein Beispiel zu bekommen, wollen wir zunächst einen Typ von Äquivalenzen weit genug dafür verstehen:

Bemerkung 2.2.6
Es gilt $\mathbf{2} \simeq (\mathbf{2} \simeq \mathbf{2})$ (mit Funktionsextensionalität).

Beweis Zunächst geben wir der Äquivalenz, die die beiden Elemente von $\mathbf{2}$ vertauscht den Namen s:

$$\begin{aligned} s &:\mathbf{2} \simeq \mathbf{2} \\ s(0_\mathbf{2}) &:\equiv 1_\mathbf{2} \\ s(1_\mathbf{2}) &:\equiv 0_\mathbf{2} \end{aligned}$$

Damit können wir Elemente von $\mathbf{2}$ auf Äquivalenzen abbilden:

$$\begin{aligned} \varphi &:\mathbf{2} \to (\mathbf{2} \simeq \mathbf{2}) \\ \varphi(0_\mathbf{2}) &:\equiv \mathrm{id}_\mathbf{2} \\ \varphi(1_\mathbf{2}) &:\equiv s \end{aligned}$$

Für die Umkehrabbildung wählen wir:

$$\begin{aligned} \psi &:(\mathbf{2} \simeq \mathbf{2}) \to \mathbf{2} \\ \psi &:\equiv f \mapsto f(0_\mathbf{2}) \end{aligned}$$

Es gilt per $\mathrm{ind}_\mathbf{2}$ durch die Urteilsgleichheiten $\psi\varphi(0_\mathbf{2}) \equiv 0_\mathbf{2}$ und $\psi\varphi(1_\mathbf{2}) \equiv 1_\mathbf{2}$ bereits $\psi \circ \varphi \sim \mathrm{id}$. Also ist noch zu zeigen:

$$\prod_{f:\mathbf{2}\simeq\mathbf{2}} \varphi(\psi(f)) = f$$

Oder nach auswerten von ψ und mit Funktionsextensionalität:

$$\prod_{f:\mathbf{2}\simeq\mathbf{2}} \prod_{x:\mathbf{2}} \varphi(f(0_\mathbf{2}))(x) = f(x)$$

Aus Beispiel 1.4.2 wissen wir, dass:

$$t : \prod_{x:\mathbf{2}} (x = 0_\mathbf{2}) \sqcup (x = 1_\mathbf{2})$$

womit wir die Fallunterscheidung "$f(0)$ ist $1_\mathbf{2}$ oder $0_\mathbf{2}$" umsetzen können:

$$f \mapsto x \mapsto t_{f(0_\mathbf{2})} : \prod_{f:\mathbf{2}\simeq\mathbf{2}} \prod_{x:\mathbf{2}} (f(0_\mathbf{2}) = 0_\mathbf{2}) \sqcup (f(0_\mathbf{2}) = 1_\mathbf{2})$$

Die beiden Fälle können wir abarbeiten, indem wir Funktionen auf den einzelnen Faktoren, in unseren Zieltyp angeben, wobei wir gleichzeitig mit $\mathrm{ind}_\mathbf{2}$ eine Fallunterscheidung über x machen. Insgesamt müssen wir also noch konstruieren:

$$\begin{aligned} F_{00} &: f(0_\mathbf{2}) = 0_\mathbf{2} \to \varphi(f(0_\mathbf{2}))(0_\mathbf{2}) = f(0_\mathbf{2}) \\ F_{01} &: f(0_\mathbf{2}) = 0_\mathbf{2} \to \varphi(f(0_\mathbf{2}))(1_\mathbf{2}) = f(1_\mathbf{2}) \\ F_{10} &: f(0_\mathbf{2}) = 1_\mathbf{2} \to \varphi(f(0_\mathbf{2}))(0_\mathbf{2}) = f(0_\mathbf{2}) \\ F_{11} &: f(0_\mathbf{2}) = 1_\mathbf{2} \to \varphi(f(0_\mathbf{2}))(1_\mathbf{2}) = f(1_\mathbf{2}) \end{aligned}$$

Denn damit haben wir dann:

$$f \mapsto \mathrm{ind}_\mathbf{2}(\ldots, \mathrm{ind}_\sqcup(\ldots, F_{00}, F_{10})(t_{f(0_\mathbf{2})}), \mathrm{ind}_\sqcup(\ldots, F_{01}, F_{11})(t_{f(0_\mathbf{2})})) : \prod_{f:\mathbf{2}\simeq\mathbf{2}} \prod_{x:\mathbf{2}} \varphi(f(0_\mathbf{2}))(x) = f(x)$$

Die Funktionen F_{00} und F_{10} können mit Gruppoidgestzen konstruiert werden. Für die übrigen müssen wir verwenden, dass $0_\mathbf{2} \neq 1_\mathbf{2}$. Etwa für F_{01} nehmen wir also $p : f(0_\mathbf{2}) = 0_\mathbf{2}$ an und verwenden wieder die Fallunterscheidung für die Gleichheit in $\mathbf{2}$:

$$(f(1_\mathbf{2}) = 0_\mathbf{2}) \sqcup (f(1_\mathbf{2}) = 1_\mathbf{2})$$

Es geht also wieder darum, zwei Funktionen zu konstruieren. Für $f(1_\mathbf{2}) = 1_\mathbf{2})$ geht das wieder mit den üblichen Methoden, für die andere Gleichheit, können wir zusammen mit p zeigen:

$$f(1_\mathbf{2}) = f(0_\mathbf{2})$$

und darauf die Inverse von f anwenden um $1_\mathbf{2} = 0_\mathbf{2}$ zu erhalten. Nun ist $0_\mathbf{2} \neq 1_\mathbf{2}$ eine Abbildung von $1_\mathbf{2} = 0_\mathbf{2}$ nach $\emptyset$. Da es von $\emptyset$ eine Abbildung in jeden beliebigen Typ gibt, haben wir insgesamt auch eine Abbildung von $1_\mathbf{2} = 0_\mathbf{2}$ in die zu zeigende Behauptung. Das beweist den Fall F_{01} und F_{11} lässt sich analog zeigen. □

Beispiel 2.2.7
Mit $\mathbf{2} \simeq (\mathbf{2} \simeq \mathbf{2})$ und per Univalenz gilt:

$$f : \mathbf{2} \simeq (\mathbf{2} =_\mathcal{U} \mathbf{2})$$

Also: $f(0_\mathbf{2}) \neq f(1_\mathbf{2})$.

Bemerkung 2.2.8
Seien $A, B, C : \mathcal{U}$, $f : A \simeq B$ und $g : B \simeq C$. Es gelten:
(a) $\mathrm{ua}(\mathrm{id}_A) = \mathrm{refl}_A$

(b) $\mathrm{ua}(f \circ g) = \mathrm{ua}(f) \cdot \mathrm{ua}(g)$

(c) $\mathrm{ua}(f^{-1}) = \mathrm{ua}(f)^{-1}$

Beweis (a) Es gilt $\mathrm{tr}_{\mathrm{id}_\mathcal{U}}(\mathrm{refl}_A) = \mathrm{id}_A$ und damit $\mathrm{ua}(\mathrm{id}_A) = \mathrm{ua}(\mathrm{tr}_{\mathrm{id}_\mathcal{U}}(\mathrm{refl}_A)) = \mathrm{refl}_A$.

(b) Seien $p :\equiv \mathrm{ua}(f)$ und $q :\equiv \mathrm{ua}(g)$. Dann gilt $\mathrm{tr}_{\mathrm{id}_\mathcal{U}}(p) = f$ und $\mathrm{tr}_{\mathrm{id}_\mathcal{U}}(q) = g$ und damit:

$$\mathrm{ua}(f \circ g) = \mathrm{ua}(\mathrm{tr}_{\mathrm{id}_\mathcal{U}}(p) \circ \mathrm{tr}_{\mathrm{id}_\mathcal{U}}(q)) = \mathrm{ua}(\mathrm{tr}_{\mathrm{id}_\mathcal{U}}(p \cdot q)) = p \cdot q = \mathrm{ua}(f) \cdot \mathrm{ua}(g)$$

(c) Sei $p :\equiv \mathrm{ua}(f)$, dann gilt:

$$\mathrm{ua}(f^{-1}) = \mathrm{ua}(\mathrm{tr}_{\mathrm{id}_\mathcal{U}}(p)^{-1}) = \mathrm{ua}(\mathrm{tr}_{\mathrm{id}_\mathcal{U}}(p^{-1})) = p^{-1} = \mathrm{ua}(f)^{-1}$$ □

Da wir nun wissen, dass Gleichheiten und Äquivalenzen dasselbe sind, können wir auch die Gleichheitsinduktion auf Äquivalenzen übertragen:

Lemma 2.2.9 (Äquivalenzinduktion)
Für $C : \prod_{A,B:\mathcal{U}} A \simeq B \to \mathcal{U}$ gibt es:

$$\mathrm{ind}_\simeq : \left(\prod_{A:\mathcal{U}} C(A, A, \mathrm{id}_A)\right) \to \prod_{A,B:\mathcal{U}} \prod_{f:A\simeq B} C(A, B, f)$$

Beweis Durch Gleichheitsinduktion ergibt sich:

$$\mathrm{ind}_= : \left(\prod_{A:\mathcal{U}} C(A, A, \mathrm{tr}_{\mathrm{id}_\mathcal{U}}(\mathrm{refl}_A))\right) \to \prod_{A,B:\mathcal{U}} \prod_{p:A=_\mathcal{U} B} C(A, B, \mathrm{tr}_{\mathrm{id}_\mathcal{U}}(p))$$ □

Wir importieren die folgende Aussage (findet man im HoTT-Book am Ende von Kapitel 4):

Fakt 2.2.10
Univalenz impliziert Funktionsextensionalität.

Im nächsten Kapitel ergibt sich ein einfacher Beweis einer stärkeren Version von Funktionsextensionalität. Wir verzichten daher im folgenden darauf, auf Verwendung von Funktionsextensionalität hinzuweisen.

2.3 Faserungen und Gleichheitssätze

Zunächst halten wir fest, dass abhängige Summen und abhängige Produkte mit äquivalenten Eingangsdaten äquivalente Typen produzieren:

Bemerkung 2.3.1
Seien $A : \mathcal{U}$ und $B, B' : A \to \mathcal{U}$. Wenn B und B' punktweise äquivalent sind, also gilt $\prod_{x:A} B(x) \simeq B'(x)$, dann gilt auch:

$$\left(\sum_{x:A} B(x)\right) \simeq \left(\sum_{x:A} B'(x)\right) \text{ und } \left(\prod_{x:A} B(x)\right) \simeq \left(\prod_{x:A} B'(x)\right)$$

Beweis Aus der punktweisen Äquivalenz $\prod_{x:A} B(x) \simeq B'(x)$ machen wir durch punktweises Anwenden von ua eine punktweise Gleichheit $\prod_{x:A} B(x) = B'(x)$ und daraus mit Funktionsextensionalität eine Gleichheit $p : B = B'$. Dann ist aber auch

$$\mathrm{ap}(\textstyle\prod, p) : \left(\prod_{x:A} B(x)\right) = \left(\prod_{x:A} B'(x)\right)$$

und damit $\simeq_{=} (\mathrm{ap}(\prod, p)) : (\prod_{x:A} B(x)) \simeq (\prod_{x:A} B'(x))$. Mit dem gleichen Argument können wir die entsprechende Aussage für abhängige Summen zeigen. □

Mit Univalenz lässt sich auch die Gleichheit von abhängigen Typen konkretisieren:

Bemerkung 2.3.2
Für $A : \mathcal{U}$ und $B, B' : A \to \mathcal{U}$ gibt es eine Äquivalenz:

$$(B = B') \simeq \left(\prod_{x:A} B(x) \simeq B'(x)\right)$$

Beweis Mit Univalenz, Bemerkung 2.3.1 und Funktionsextensionalität:

$$\left(\prod_{x:A} B(x) \simeq B'(x)\right) \simeq \left(\prod_{x:A} B(x) = B'(x)\right) \simeq (B = B')$$

□

Definition 2.3.3
Sei $A : \mathcal{U}$ und $B, B' : A \to \mathcal{U}$ zwei abhängige Typen.

(a) Eine *faserweise Abbildung* ist ein Term

$$f : \prod_{x:A} (B(x) \to B'(x))$$

(b) Für eine faserweise Abbildung gibt es eine induzierte Abbildung

$$\textstyle\sum f : \left(\sum_{x:A} B(x)\right) \to \left(\sum_{x:A} B'(x)\right)$$

gegeben durch $\sum f(x, b_x) \equiv (x, f_x(b_x))$.

Lemma 2.3.4
Seien $A : \mathcal{U}$ und $B : A \to \mathcal{U}$. Dann gilt

$$\prod_{x:A} \pi_1^{-1}(x) \simeq B(x)$$

Beweis Später. □

Mit Univalenz können wir folgern, dass für jeden Typ A die abhängigen Typen auf A dasselbe sind, wie Funktionen nach A.

Theorem 2.3.5 (Objektklassifikation)
Sei $A : \mathcal{U}$. Dann gilt:

$$\left(\sum_{B:\mathcal{U}} (B \to A)\right) \simeq (A \to \mathcal{U})$$

Beweis Einen abhängigen Typ $B : A \to \mathcal{U}$ bilden wir auf die Funktion

$$\pi_1 : \left(\sum_{x:A} B(x)\right) \to A$$

ab. Und eine Funktion $f : B \to A$ bilden wir auf den abhängigen Typ ihrer Fasern ab:

$$\mathrm{fib}_f \equiv \big((x : A) \mapsto f^{-1}(x)\big) : A \to \mathcal{U}$$

Nach Lemma 2.3.4 ist also bereits diese Konstruktion linksinvers zur vorangegangenen. Es ist also noch zu zeigen, dass sich die Fasern einer Abbildung zur Domäne aufsummieren, bzw. dass für alle $f : A \to B$ gilt:

$$e : \left(\sum_{y:B} (f^{-1}(y))\right) \simeq A$$

□

und $f \circ e = \pi_1$ gilt. Wir können mit Lemma 1.6.9 berechnen:

$$\begin{aligned}\left(\sum_{y:B} (f^{-1}(y))\right) &\simeq \sum_{y:B} \sum_{x:A} f(x) = y \\ &\simeq \sum_{x:A} \sum_{y:B} f(x) = y \\ &\simeq \sum_{x:A} \mathbf{1} \\ &\simeq A\end{aligned}$$

Lemma 2.3.6
Seien $A : \mathcal{U}$, $B, B' : A \to \mathcal{U}$ und $f : \prod_{x:A} B(x) \to B'(x)$. Für alle $y : \sum_{x:A} B'(x)$ gilt:

$$\left((\sum f)^{-1}(y)\right) \simeq f^{-1}_{\pi_1(y)}(\pi_2(y))$$

Beweis Unser erstes Ziel ist es, Abbildungen in beide Richtungen durch Gleichheitsinduktion zu definieren. Sei dazu zunächst:

$$X : \prod_{z:\sum_{x:A} B(x)} \prod_{y:\sum_{x:A} B'(x)} \prod_{p:(\sum f)(z)=y} f^{-1}_{\pi_1(y)}(\pi_2(y))$$

gegeben durch $X(z, (\sum f)(z), \mathrm{refl}_{(\sum f)(z)}) :\equiv (\pi_1(z), \mathrm{refl}_{f_{\pi_1(z)}(\pi_2(z))})$. Damit ist

$$\begin{aligned}&\varphi : \prod_{y:\sum_{x:A} B'(x)} \left((\sum f)^{-1}(y)\right) \to f^{-1}_{\pi_1(y)}(\pi_2(y)) \\ &\varphi(y, (z, p)) :\equiv X(z, y, p)\end{aligned}$$

und mit der gleichen Vorgehensweise lässt sich

$$\psi : \prod_{y:\sum_{x:A} B'(x)} f^{-1}_{\pi_1(y)}(\pi_2(y)) \to \left((\sum f)^{-1}(y)\right)$$

definieren und Homotopien, die zeigen dass φ und ψ zueinander invers sind. □

Lemma 2.3.7
Für $f : \prod_{x:A} B(x) \to B'(x)$ gilt: f ist genau dann faserweise Äquivalenz, wenn $\sum f$ Äquivalenz ist.

Beweis Nach Lemma 2.3.6 sind alle Fasern aller f_x genau dann kontrahierbar, wenn die Fasern von $\sum f$ kontrahierbar sind. □

Theorem 2.3.8 (Fundamentaler Gleichheitssatz)
Seien $A : \mathcal{U}$ und $a : A$. Für $B : A \to \mathcal{U}$ und eine faserweise Abbildung $f : \prod_{x:A}(a = x) \to B(x)$ ist f genau dann eine faserweise Äquivalenz, wenn

$$\sum_{x:A} B(x)$$

kontrahierbar ist.

Beweis Mit dem vorangegangenen Lemma wissen wir, dass f genau dann eine faserweise Äquivalenz ist, wenn

$$\sum f : \left(\sum_{x:A} a = x\right) \to \left(\sum_{x:A} B(x)\right)$$

eine Äquivalenz ist. In Lemma 1.6.9 haben wir gesehen, dass der linke Typ kontrahierbar ist, also gilt:

$$\left(\sum_{x:A} a = x\right) \simeq \mathbf{1}$$

Wenn nun auch $\mathbf{1} \simeq \sum_{x:A} B(x)$ gilt, dann ist die Abbildung $\sum f$ bereits eine Äquivalenz, also auch f eine faserweise Äquivalenz. □

Beispiel 2.3.9
Sei $\mathrm{Eq}_{\mathbf{2}} : \mathbf{2} \to \mathbf{2} \to \mathcal{U}$ gegeben durch Rekursion in beiden Argumenten:

$$\begin{aligned}
\mathrm{Eq}_{\mathbf{2}}(0_{\mathbf{2}}, 0_{\mathbf{2}}) &:\equiv \mathbf{1} \\
\mathrm{Eq}_{\mathbf{2}}(0_{\mathbf{2}}, 1_{\mathbf{2}}) &:\equiv \emptyset \\
\mathrm{Eq}_{\mathbf{2}}(1_{\mathbf{2}}, 0_{\mathbf{2}}) &:\equiv \emptyset \\
\mathrm{Eq}_{\mathbf{2}}(1_{\mathbf{2}}, 1_{\mathbf{2}}) &:\equiv \mathbf{1}
\end{aligned}$$

Mit Theorem 2.3.8 können wir zeigen, dass

$$\prod_{x,y:\mathbf{2}} (x =_{\mathbf{2}} y) \simeq \mathrm{Eq}_{\mathbf{2}}(x, y)$$

indem wir für jedes $x : \mathbf{2}$ zeigen, dass $\sum_{y:\mathbf{2}} \mathrm{Eq}_{\mathbf{2}}(x, y)$ kontrahierbar ist. Per **2**-Induktion reicht es das für $x \equiv 0_{\mathbf{2}}, 1_{\mathbf{2}}$ zu zeigen. Für

$$\sum_{y:\mathbf{2}} \mathrm{Eq}_{\mathbf{2}}(0_{\mathbf{2}}, y)$$

wählen wir als Kontraktionszentrum $(0_{\mathbf{2}}, *)$. Nun müssen wir zeigen, dass jeder andere Punkt gleich diesem Kontraktionszentrum ist, also mit $\sum$-Induktion:

$$\prod_{y:\mathbf{2}} \prod_{e:\mathrm{Eq}_{\mathbf{2}}(0_{\mathbf{2}}, y)} (0_{\mathbf{2}}, *) = (y, e)$$

Mit Anwenden von **2**-Induktion auf y reicht es also, die beiden Fälle $\prod_{e:\mathrm{Eq}_{\mathbf{2}}(0_{\mathbf{2}}, 0_{\mathbf{2}})} (0_{\mathbf{2}}, *) = (0_{\mathbf{2}}, e)$ und $\prod_{e:\mathrm{Eq}_{\mathbf{2}}(0_{\mathbf{2}}, 1_{\mathbf{2}})} (0_{\mathbf{2}}, *) = (1_{\mathbf{2}}, e)$ abzuhandeln. Im ersten Fall kann **1**-Induktion angewandt werden und $\mathrm{refl}_{(0_{\mathbf{2}}, *)}$ ist eine Lösung. Im zweiten Fall ist $\emptyset$-Induktion eine Lösung.
Analog lässt sich zeigen, dass $\sum_{y:\mathbf{2}} \mathrm{Eq}_{\mathbf{2}}(1_{\mathbf{2}}, y)$, was den Beweis beendet.

Es ist also insbesondere **2** ein 0-Typ bzw. eine Menge.

Wir wollen nun noch Lemma 2.3.4 beweisen. Dazu brauchen wir das folgende alternative Induktionsprinzip für Gleichheit:

Lemma 2.3.10 (Pfadinduktion mit Basispunkt)
Seien $A : \mathcal{U}$ und $x : A$ fest gewählt. Für einen abhängigen Typen $C : \prod_{y:A} x = y \to \mathcal{U}$ reicht es, ein $c : C(x, \mathrm{refl}_x)$ zu konstruieren, um eine abhängige Funktion $\prod_{y:A} \prod_{p:x=y} C(y, p)$ zu erhalten.

Beweis Sei $c : C(x, \mathrm{refl}_x)$. Dann gibt es den abhängigen Typen

$$C' :\equiv (z : \sum_{y:A} x = y) \mapsto C(\pi_1(z), \pi_2(z))$$

Wir wissen, dass die Basis $\sum_{y:A} x = y$ kontrahierbar ist, also gibt es Gleichheiten $q_{y,p} : (x, \mathrm{refl}_x) = (y, p)$, für alle $y : A$ und $p : x = y$ und damit

$$\mathrm{tr}_C(q_{y,p}) : C(x, \mathrm{refl}_x) \to C(y, p)$$

Also

$$y \mapsto (p : x = y) \mapsto \mathrm{tr}_C(q_{y,p})(c) : \prod_{y:A} \prod_{p:x=y} C(y, p)$$

□

Damit können wir zeigen:

Lemma 2.3.11
Seien $A : \mathcal{U}$ und $B : A \to \mathcal{U}$. Für $C : (\sum_{x:A} B(x)) \to \mathcal{U}$ gilt:

$$\left(\sum_{y:\sum_{x:A} B(x)} C(y)\right) \simeq \sum_{x:A} \sum_{b:B(x)} C((x,b))$$

Beweis Aus Übungsaufgabe 1 von Blatt 6 wissen wir, dass es einen Term gibt (mit $p_{(x,b)} \equiv \mathrm{refl}_{(x,b)}$):

$$p : \prod_{z:\sum_{x:A} B(x)} z = (\pi_1(z), \pi_2(z))$$

Es gibt die beiden folgenden Kandidaten für zueinander inversen Abbildungen:

$$\begin{aligned}
\varphi &: \left(\sum_{y:\sum_{x:A} B(x)} C(y)\right) \to \sum_{x:A} \sum_{b:B(x)} C((x,b)) \\
\varphi((y,c)) &\coloneqq (\pi_1(y), (\pi_2(y), \mathrm{tr}_C(p_y)(c))) \\
\psi &: \left(\sum_{x:A} \sum_{b:B(x)} C((x,b))\right) \to \left(\sum_{y:\sum_{x:A} B(x)} C(y)\right) \\
\psi((x,(b,c))) &\coloneqq ((x,b),c)
\end{aligned}$$

Damit gilt $\varphi \circ \psi \sim \mathrm{id}$ bereits punktweise urteilsmäßig, es ist also noch zu zeigen, dass auch $\psi \circ \varphi \sim \mathrm{id}$ gilt. Für $y : \sum_{x:A} B(x)$ und $c : C(y)$ gilt:

$$\begin{aligned}
\psi(\varphi(y,c)) &= \psi(\pi_1(y), (\pi_2(y), \mathrm{tr}_C(p_y)(c))) \\
&= ((\pi_1(y), \pi_2(y)), \mathrm{tr}_C(p_y)(c))
\end{aligned}$$

Letzteres ist allerdings mit $\sum_=$ und p_y gleich zu (y,c). □

Damit können wir nun Lemma 2.3.4 beweisen:

Beweis (von Lemma 2.3.4) Seien $A : \mathcal{U}$, $B : A \to \mathcal{U}$ und $x : A$. Für $\pi_1 : \sum_{x:A} B(x) \to A$ gilt:

$$\begin{aligned}
\pi_1^{-1}(x) &\equiv \sum_{y:\sum_{x':A} B(x')} \pi_1(y) = x \\
&\simeq \sum_{x':A} \sum_{b:B(x')} \pi_1((x',b)) = x \\
&\equiv \sum_{x':A} \sum_{b:B(x')} x' = x \\
&\simeq \sum_{x':A} \sum_{p:x'=x} B(x') \\
&\simeq B(x)
\end{aligned}$$

2.4 Gleichheit natürlicher Zahlen

Definition 2.4.1
$\mathrm{Eq}_{\mathbb{N}} : \mathbb{N} \to \mathbb{N} \to \mathcal{U}$ definieren wir durch doppelte Rekursion wie folgt:

$$\begin{aligned}
\mathrm{Eq}_{\mathbb{N}}(0_{\mathbb{N}}, 0_{\mathbb{N}}) &\coloneqq \mathbf{1} \\
\mathrm{Eq}_{\mathbb{N}}(0_{\mathbb{N}}, \mathrm{succ}_{\mathbb{N}}(k)) &\coloneqq \emptyset \\
\mathrm{Eq}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(n), 0_{\mathbb{N}}) &\coloneqq \emptyset \\
\mathrm{Eq}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(n), \mathrm{succ}_{\mathbb{N}}(k)) &\coloneqq \mathrm{Eq}_{\mathbb{N}}(n,k)
\end{aligned}$$

Theorem 2.4.2 (Gleichheit natürlicher Zahlen)
Die faserweise Abbildung

$$\prod_{n,l:\mathbb{N}} n = l \to \mathrm{Eq}_{\mathbb{N}}(n,l)$$

gegeben durch $\mathrm{refl}_n \mapsto *$ ist eine faserweise Äquivalenz.

Beweis Nach Theorem 2.3.8 reicht es zu zeigen, dass für jedes $n : \mathbb{N}$ der Typ

$$\sum_{l:\mathbb{N}} \mathrm{Eq}_{\mathbb{N}}(n, l)$$

kontrahierbar ist. Als Kontraktionszentrum wählen wir $(n, *)$. Nach Induktion über n, l und Ausschluss aller Fälle mit leerer Domäne bleiben noch zu konstruieren:

$$\prod_{e:\mathrm{Eq}_{\mathbb{N}}(0,0)} (0, e) = (0, *)$$

$$\prod_{e:\mathrm{Eq}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(n),\mathrm{succ}_{\mathbb{N}}(l))} (\mathrm{succ}_{\mathbb{N}}(l), e) = (\mathrm{succ}_{\mathbb{N}}(n), *)$$

Im ersten Fall lässt sich die Gleichheit stets durch die Kontrahierbarkeit von $\mathbf{1}$ konstruieren. Im zweiten Fall dürfen wir die Induktionshypothese verwenden:

$$\mathrm{IH} : \prod_{e:\mathrm{Eq}_{\mathbb{N}}(n,l)} (l, e) = (n, *)$$

Um etwas damit anfangen zu können, definieren wir uns folgenden Funktion:

$$f : \left(\sum_{n:\mathbb{N}} \mathrm{Eq}_{\mathbb{N}}(n, l)\right) \to \left(\sum_{n:\mathbb{N}} \mathrm{Eq}_{\mathbb{N}}(n, \mathrm{succ}_{\mathbb{N}}(l))\right)$$
$$f((n, e)) :\equiv (n + 1, e)$$

Damit gilt dann für $e : \mathrm{Eq}_{\mathbb{N}}(\mathrm{succ}_{\mathbb{N}}(n), \mathrm{succ}_{\mathbb{N}}(l))$:

$$f(\mathrm{IH}_e) : (f((l, e)) = f((n, *))) \equiv ((\mathrm{succ}_{\mathbb{N}}(l), e) = (\mathrm{succ}_{\mathbb{N}}(n), *)) \qquad \square$$

Bemerkung 2.4.3
$\mathbb{N}$ ist ein 0-Typ.

Beweis Wir wissen dass die Typen $\emptyset$ und $\mathbf{1}$ Aussagen sind durch Beispiel 1.6.6 und Bemerkung 1.6.7. Mit $\mathbb{N}$-Induktion ist also $\mathrm{Eq}_{\mathbb{N}}(n, k)$ für alle $n, k \in \mathbb{N}$ eine Aussage. Nach Definition ist ein Typ ein 0-Typ oder eine Menge, wenn alle Gleichheitstypen Aussagen sind, also sind wir mit Theorem 2.4.2 fertig. $\square$

2.5 Gleichheit algebraischer Strukturen

Wir wollen in diesem Abschnitt am Beispiel der Halbgruppen sehen, dass Gleichheiten zwischen algebraischen Objekten genau den Isomorphismen entsprechen. Da es sich etwa beim Typ der Halbgruppen um eine abhängige Summe handeln wird und wir die Gleichheitstypen vollständig verstehen wollen, werden wir zunächst die Chrakterisierung von Gleichheitstypen in abhängigen Summen abschließen.

Lemma 2.5.1 (Gleichheit in abhängigen Summen)
Seien $A : \mathcal{U}$ und $B : A \to \mathcal{U}$. Für $a, a' : A$ und $b : B(a), b' : B(a')$ gilt:

$$((a, b) = (a', b')) \simeq \sum_{p:a=a'} \mathrm{tr}_B(p)(b) = b'$$

Beweis Wir verwenden den Gleichheitssatz, Theorem 2.3.8. Sei dazu der abhängige Typ $\mathrm{Eq}_{\sum} : (\sum_{x:A} B(x)) \to (\sum_{x:A} B(x)) \to \mathcal{U}$ gegeben durch $\sum$-Induktion, also für $a, a' : A$ und $b : B(a), b' : B(a')$:

$$\mathrm{Eq}_{\sum}((a, b), (a', b')) :\equiv \sum_{p:a=a'} \mathrm{tr}_B(p)(b) = b'$$

und weiter $r : \prod_{x:\sum_{a:A} B(a)} \mathrm{Eq}_{\sum}(x, x)$ gegeben durch:

$$r((a, b)) :\equiv (\mathrm{refl}_a, \mathrm{refl}_b)$$

Damit gibt es eine durch Gleichheitsinduktion definierte Funktion

$$f : \prod_{x,y:\sum_{a:A} B(a)} x = y \to \mathrm{Eq}_{\sum}(x, y)$$

Nach dem Gleichheitssatz reicht es nun also zu zeigen, dass für $x : \sum_{a:A} B(a)$ der Typ

$$\sum_{y:\sum_{a:A} B(a)} \mathrm{Eq}_{\sum}(x, y)$$

kontrahierbar ist. Es gilt für $x \equiv (a, b)$:

$$\begin{aligned}
\sum_{y:\sum_{a:A} B(a)} \mathrm{Eq}_{\sum}((a,b), y) &\simeq \sum_{a':A} \sum_{b':B(a')} \mathrm{Eq}_{\sum}((a,b),(a',b')) \\
&\equiv \sum_{a':A} \sum_{b':B(a')} \sum_{p:a=a'} \mathrm{tr}_B(p)(b) = b' \\
&\simeq \sum_{a':A} \sum_{p:a=a'} \sum_{b':B(a')} \mathrm{tr}_B(p)(b) = b' \\
&\simeq \sum_{a':A} \sum_{p:a=a'} \mathbf{1} \\
&\simeq \mathbf{1}
\end{aligned}$$

□

Definition 2.5.2
Set $\coloneqq \sum_{A:\mathcal{U}} \mathrm{isSet}(A)$

Definition 2.5.3
Sei $A : \mathcal{U}$. Der Typ der *Halbgruppenstrukturen* auf A ist gegeben durch:

$$\mathrm{HalbGrp}(A) \coloneqq \sum_{\_\cdot\_:A\to A\to A} \prod_{x,y,z:A} (x \cdot y) \cdot z = x \cdot (y \cdot z)$$

Die Elemente $H : \sum_{A:\mathrm{Set}} \mathrm{HalbGrp}(A)$, heißen *Halbgruppen.*

Definition 2.5.4
Ein *Isomorphismus* von Halbgruppen $(H, \mathrm{str}) \to (H', \mathrm{str}')$ ist eine Äquivalenz $f : H \to H'$, sodass für alle $x, y : H$ gilt:

$$f(x \cdot y) = f(x) \cdot' f(y)$$

wobei $\cdot$ die Multiplikation auf H und $\cdot'$ die auf H' ist. Wir bezeichnen mit $\mathrm{Isom}((H, \mathrm{str}), (H', \mathrm{str}'))$ den Typ der Isomorphismen zwischen zwei Halbgruppen.

Theorem 2.5.5 (Gleichheit von Halbgruppen)
Seien $(H, \mathrm{str}), (H', \mathrm{str}) : \sum_{A:\mathcal{U}} \mathrm{HalbGrp}(A)$. Es gilt:

$$((H, \mathrm{str}) = (H', \mathrm{str})) \simeq \mathrm{Isom}((H, \mathrm{str}), (H', \mathrm{str}'))$$

Beweis Mit Lemma 2.5.1 folgt:

$$((H, \mathrm{str}) = (H', \mathrm{str}')) \simeq \sum_{p:H=H'} \mathrm{tr}_{\mathrm{HalbGrp}}(p)(\mathrm{str}) = \mathrm{str}'$$

Das heißt, wir müssen nur noch klären, dass diese Summe äquivalent zum Typ der Isomorphismen ist. Dazu berechnen wir den Transport in HalbGrp. Sei also $e : A \simeq B$ und $p \coloneqq \mathrm{ua}(e)$, dann erhalten wir eine Äquivalenz

$$\mathrm{tr}_{\mathrm{HalbGrp}}(p) : \mathrm{HalbGrp}(A) \to \mathrm{HalbGrp}(B)$$

und wir können durch Raten und anschließendem Bestätigen durch Gleichheitsinduktion berechnen, was dieser Transport macht. Sei $(\_\cdot\_, \cdot\mathrm{assoc}) : \mathrm{HalbGrp}(A)$. Wir raten zunächst, dass die Multiplikation mit e prä- und postkomponiert wird, die Multiplikation des Bildes also für $x, y : B$ gegeben ist durch:

$$x \cdot' y \coloneqq e(e^{-1}(x) \cdot e^{-1}(y))$$

Als Assoziativitätszeugen $\cdot'\mathrm{assoc}$ nehmen wir die Ergebnisgleichheit der folgenden Rechnung:

$$\begin{aligned}
(x \cdot' y) \cdot' z &= e(e^{-1}e(e^{-1}(x) \cdot e^{-1}(y)) \cdot e^{-1}(z)) \\
&= e((e^{-1}(x) \cdot e^{-1}(y)) \cdot e^{-1}(z)) \\
&= e(e^{-1}(x) \cdot (e^{-1}(y) \cdot e^{-1}(z))) \\
&= e(e^{-1}(x) \cdot e^{-1}(e(e^{-1}(y) \cdot e^{-1}(z)))) \\
&= x \cdot' (y \cdot' z)
\end{aligned}$$

Die Abbildung e ist damit ein Isomorphismus. Nun können wir für diese Konstruktion sehen:

$$\mathrm{tr}_{\mathrm{HalbGrp}}(\mathrm{ua}(\mathrm{id}))((\_ \cdot \_, \cdot\mathrm{assoc})) = (\_ \cdot' \_, \cdot'\mathrm{assoc}) \qquad \square$$

3 Homotopietheorie

3.1 Höhere Induktive Typen

Höhere induktive Typen haben neben Konstruktoren der Bauart, die wir bereits von den Induktiven Typen kennen, sogenannte *höhere Konstruktoren*, deren Werte in Gleichheitstypen des höheren induktiven Typs liegen. Später werden wir auch höhere induktive Typen mit höheren Konstruktoren, die in iterierten Gleichheitstypen liegen, sehen. Zunächst wird der Typ S^1 zentral sein. Dieser hat wie der Typ $\mathbf{1}$ einen Punkt-Konstruktor $* : S^1$ und zusätzlich einen höheren Konstruktor $l : * =_{S^1} *$.

Eine Rekursionsregel ist leicht formuliert: Um eine Funktion $f : S^1 \to A$ zu definieren, muss man $f(*) : A$ und $f(l) : f(*) =_A f(*)$ vorgeben. Das Induktionsprinzip ist etwas unhandlicher und wir wollen zunächst ein paar Definition machen, die uns die Formulierung erleichtern und schließlich nach der Einführung der neuen Regeln das Rekursionsprinzip aus dem Induktionsprinzip folgern.

In Lemma 2.5.1 haben wir festgestellt, dass ein Teil einer Gleichheit in einer abhängigen Summe von der Form

$$\mathrm{tr}_B(p)(b) = b'$$

ist. Das werden wir jetzt als allgemeine Definition von Gleichheiten in abhängigen Typen verwenden.

Definition 3.1.1 (Abhängige Gleichheit)
Seien $A : \mathcal{U}$, und $B : A \to \mathcal{U}$. Seien weiter $x, y : A$, $b : B(x)$ und $b' : B(y)$.

(a) Sei $p : x =_A y$. Ein *abhängier Pfad* oder eine *abhängige Gleichheit* zwischen b und b' ist ein Element des Typs

$$\left(b =_p^B b'\right) :\equiv (\mathrm{tr}_B(p)(b) = b')$$

(b) Sei $s : \prod_{x:A} B(x)$ eine abhängige Funktion. Die folgende Funktion ist die Anwendung eine abhängigen Funktion auf eine Gleichheit:

$$\mathrm{apd}(s) : \prod_{p:x=_A y} s(x) =_p^B s(y)$$

und festgelegt durch $\mathrm{apd}(s, \mathrm{refl}_x) :\equiv \mathrm{refl}_b$. Wir schreiben auch $s(p)$ für $\mathrm{apd}(s, p)$.

Damit können wir die folgenden Regeln für den induktiven Einheitskreis leichter formulieren. Zunächst wollen wir noch die abhängige Variante mit der bekannten Anwendung von Funktionen auf Gleichheiten vergleichen:

Bemerkung 3.1.2
Seien $A, B : \mathcal{U}$. Dann gilt:

(a) Für $x, y : A$, $p : x =_A y$ und alle $b : B$ erhalten wir eine abhängige Gleichheit

$$\mathrm{tconst}_{p,b} : \mathrm{tr}_{\_\mapsto B}(p)(b) = b$$

durch Induktion über p und $\mathrm{tconst}_{\mathrm{refl}_x, b} :\equiv \mathrm{refl}_b$.

(b) Für $f : A \to B$ und $p : x =_A y$ gilt $\mathrm{apd}(f, p) = \mathrm{tconst}_{p, f(x)} \cdot \mathrm{ap}(f, p)$.

Beweis (a) Bereits erledigt.

(b) Induktion über p. $\square$

Anders als bei den Konstruktoren induktiver Typen, werden wir bei höheren Konstruktoren keine urteilsmäßige Gleichheit für Berechnungen fordern, sondern nur Gleichheit.

Regeln 3.1.3
Es gibt einen Typ $S^1 : \mathcal{U}$ den wir den (höheren induktiven) *Kreis* nennen. S^1 ist der höhere induktive Typ mit den Konstruktoren:

$$* : S^1$$
$$l : * =_{S^1} *$$

Das heißt für jeden abhängigen Typ $B : S^1 \to \mathcal{U}$ reicht es $b : B(*)$ und $b_l : b =_l^B b$ vorzugeben, um eine abhängige Funktion $s : \prod_{x:S^1} B(x)$ zu definieren. Es gibt also eine Funktion:

$$\mathrm{ind}_{S^1} : \prod_{B:S^1\to\mathcal{U}} \prod_{b:B(*)} \left((b =_l^B b) \to \prod_{x:S^1} B(x) \right)$$

die *Kreisinduktion*. Es gelten folgende Berechnungsregeln:

$$\begin{aligned} \mathrm{ind}_{S^1}(B, b, b_l)(*) &\equiv b \\ \mathrm{apd}(\mathrm{ind}_{S^1}(B, b, b_l), l) &\equiv \mathrm{ind}_{S^1}(B, b, b_l)(l) = b_l \end{aligned}$$

Wir werden abhängige Funktionen $s : \prod_{x:S^1} B(x)$ auch durch Fallunterscheidung wie folgt angeben:

$$\begin{aligned} s(*) &:\equiv b \\ s(l) &:= b_l \end{aligned}$$

Bemerkung 3.1.4
Wie bereits bei den induktiven Typen, werden wir auch hier darauf verzichten, ein allgemeines Schema anzugeben. Unter Homotopietypentheorie versteht man eine Typentheorie wie soweit eingeführt, in der jeder sinnvolle höhere induktive Typ existiert. Wir werden lediglich Beispiele von sinnvollen höheren induktiven Typen kennenlernen.

Wir wollen zunächst das Rekursionprinzip für den Kreis herleiten.

Bemerkung 3.1.5
Für $A : \mathcal{U}$, $a : A$ und $a_l : a =_A a$ gibt es stets eine Funktion $\mathrm{rec}_{S^1}(A, a, a_l) : S^1 \to A$ mit

$$\begin{aligned} \mathrm{rec}_{S^1}(A, a, a_l)(*) &\equiv a \\ \mathrm{rec}_{S^1}(A, a, a_l)(l) &= a_l \end{aligned}$$

Beweis Wir verwenden Kreisinduktion für den konstanten abhängigen Typ $\_ \mapsto A : S^1 \to \mathcal{U}$. Es gibt eine Äquivalenz

$$(a =_l a) \simeq (a =_A a)$$

durch Linkskonkatenation mit $\mathrm{tconst}_{l,a}^{-1}$. Wir verwenden die Inverse, um aus $a_l : a =_A a$ die Gleichheit $\mathrm{tconst}_{l,a} \cdot a_l : a =_l a$ zu produzieren. Damit gilt für

$$\mathrm{rec}_{S^1}(A, a, a_l) :\equiv \mathrm{ind}_{S^1}(\_ \mapsto A, a, \mathrm{tconst}_{l,a} \cdot a_l)$$

nach bemerkung 3.1.2:

$$\begin{aligned} \mathrm{ap}(\mathrm{rec}_{S^1}(A, a, a_l), l) &\equiv \mathrm{ap}(\mathrm{ind}_{S^1}(\_ \mapsto A, a, \mathrm{tconst}_{l,a} \cdot a_l), l) \\ &= \mathrm{tconst}_{l,a}^{-1} \cdot \mathrm{apd}(\mathrm{ind}_{S^1}(\_ \mapsto A, a, \mathrm{tconst}_{l,a} \cdot a_l), l) \\ &= \mathrm{tconst}_{l,a}^{-1} \cdot \mathrm{tconst}_{l,a} \cdot a_l \\ &= a_l \end{aligned}$$

Nach Definition gilt außerdem $\mathrm{rec}_{S^1}(A, a, a_l)(*) \equiv a$. □

Bevor wir den Kreis weiter kennenlernen, betrachten wir noch einen weiteren, sehr ähnlichen höheren induktiven Typ.

Regeln 3.1.6
Es gibt einen Typ $I : \mathcal{U}$, das *Intervall* mit den folgenden Konstruktoren:

$$\begin{aligned} 0_I &: I \\ 1_I &: I \\ s &: 0_I =_I 1_I \end{aligned}$$

Daraus ergibt sich das folgende Induktionsprinzip, die *Intervallinduktion*:

$$\mathrm{ind}_I : \prod_{B:I\to\mathcal{U}} \prod_{b_0:B(0_I)} \prod_{b_1:B(1_I)} b_0 =_s^B b_1 \to \prod_{x:I} B(x)$$

mit Berechnungsregeln:

$$\begin{aligned}\mathrm{ind}_I(B, b_0, b_1, b_s)(0_I) &\equiv b_0\\ \mathrm{ind}_I(B, b_0, b_1, b_s)(1_I) &\equiv b_1\\ \mathrm{apd}(\mathrm{ind}_I(B, b_0, b_1, b_s), s) \equiv \mathrm{ind}_I(B, b_0, b_1, b_s)(s) &= b_s\end{aligned}$$

Bemerkung 3.1.7
Das Intervall I ist kontrahierbar.

Beweis Als Kontraktionszentrum wählen wir 0_I. Nun wollen wir zeigen:

$$\prod_{x:I} 0_I = x$$

Mit Intervallinduktion reicht es dafür festzustellen, dass $\mathrm{refl} : 0_I = 0_I$ und $s : 0_i = 1_I$ gelten und zu zeigen:

$$\mathrm{refl} =_s s$$

oder anders formuliert: $\mathrm{tr}_{0_I =\_}(s)(\mathrm{refl}) = s$. Das haben wir aber bereits allgemein geklärt in Lemma 1.4.14. □

Trotzdem können wir mit dem Intervall etwas neues zeigen. Davor brauchen wir noch das Rekursionsprinzip.

Lemma 3.1.8 (Intervallrekursion und ihre Eindeutigkeit)
Sei $A : \mathcal{U}$.

(a) Für $a, a' : A$ und $a_s : a =_A a'$ gibt es

$$\mathrm{rec}_I(A, a, a', a_s) : I \to A$$

mit $\mathrm{rec}_I(A, a, a', a_s)(0_I) \equiv a$, $\mathrm{rec}_I(A, a, a', a_s)(1_I) \equiv a'$ und $\mathrm{rec}_I(A, a, a', a_s)(s) = a_s$.

(b) Für $f : I \to A$ gilt:

$$f = \mathrm{rec}_I(A, f(0_I), f(1_I), f(s))$$

Beweis (a) Wie bei der Kreisrekursion.

(b) Wegen Funktionsextensionalität genügt es, $\prod_{x:I} f(x) = \mathrm{rec}_I(A, f(0_I), f(1_I), f(s))(x)$ zu zeigen. Dazu verwenden wir I-Induktion:

- $\mathrm{refl}_{f(0_I)} : f(0_I) = \underbrace{\mathrm{rec}_I(A, f(0_I), f(1_I), f(s))(0_I)}_{\overset{(a)}{\equiv} f(0_I)}$
- $\mathrm{refl}_{f(1_I)} : f(1_I) = \underbrace{\mathrm{rec}_I(A, f(1_I), f(1_I), f(s))(1_I)}_{\overset{(a)}{\equiv} f(1_I)}$
- Es verbleibt also, $\mathrm{refl}_{f(0_I)} =_s^{f(\_)=r(\_)} \mathrm{refl}_{f(1_I)}$ zu zeigen. Dabei ist

$$(\mathrm{refl}_{f(0)} =_s^{f(\_)=r(\_)} \mathrm{refl}_{f(1)}) \overset{\text{ÜA}}{\equiv} (f(s)^{-1} \cdot \mathrm{refl}_{f(0)} \cdot \underbrace{\mathrm{rec}_I(A, f(0), f(1), f(s))(s)}_{\overset{(a)}{\equiv} f(s)} = \mathrm{refl}_{f(1)}).$$

  Damit beweist $\mathrm{refl}_{\mathrm{refl}_{f(1)}}$ die Aussage. □

Zunächst erweitern wir die Definition von Homotopien auf abhängige Typen:

Definition 3.1.9
Seien $A : \mathcal{U}$, $B : A \to \mathcal{U}$ und $f, g : \prod_{x:A} B(x)$. Eine *abhängige Homotopie* von f nach g ist ein Term von

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

Weiter verwenden wir in diesem Abschnitt die Funktion

$$\mathrm{hap} :\equiv (p : f = g) \mapsto (x : A) \mapsto \mathrm{ap}(f \mapsto f(x), p) : f = g \to f \sim g$$

Theorem 3.1.10 (Abhängige Funktionsextensionalität)
Seien $A : \mathcal{U}$ und $B : A \to \mathcal{U}$. Dann gilt für abhängige Funktionen $f, g : \prod_{x:A} B(x)$:

(a) Es gibt eine Funktion $\varphi : f \sim g \to f = g$ und es gilt $\mathrm{hap}(\varphi(H)) = H$ für jede abhängige Homotopie $H : f \sim g$.

(b) Die Funktion
$$\mathrm{hap} : f = g \to f \sim g$$
ist eine Äquivalenz.

Beweis (a) Zunächst konstruieren wir für $H : \prod_{x:A} f(x) = g(x)$ eine Gleichheit $p_H : f = g$. Für $x : A$ können wir die Gleichheit $f(x) = g(x)$ durch eine Abbildung $I \to B(x)$ ersetzen und damit H durch die Abbildung
$$\tilde{H} :\equiv (x : A) \mapsto \mathrm{rec}_I(B(x), f(x), g(x), H_x) : \prod_{x:A} I \to B(x)$$
Nun vertauschen wir A und I:
$$\tilde{\tilde{H}} :\equiv (i : I) \mapsto (x : A) \mapsto \tilde{H}(x)(i) : I \to \prod_{x:A} B(x)$$
Und damit haben wir $p_H :\equiv \tilde{\tilde{H}}(s) : f = g$.

Nun können wir durch kommutieren von ap und $\circ$ sehen, dass gilt:
$$\begin{aligned}
\mathrm{hap}(\varphi(H)) &\equiv (x : A) \mapsto \mathrm{ap}(f \mapsto f(x), \varphi(H)) \\
&\equiv (x : A) \mapsto \mathrm{ap}(f \mapsto f(x), \mathrm{ap}((i : I) \mapsto (y : A) \mapsto \tilde{H}(y)(i), s)) \\
&= (x : A) \mapsto \mathrm{ap}((i : I) \mapsto \tilde{H}(x)(i), s) \\
&= (x : A) \mapsto H_x
\end{aligned}$$

(b) Nun können wir den Gleichheitssatz verwenden, wenn wir es schaffen zu zeigen, dass
$$\sum_{g:\prod_{x:A} B(x)} f \sim g$$
kontrahierbar ist. Dazu berechnen wir erstmal mit Gleichheitsinduktion über $p : g = g'$, dass für den entsprechenden Transport und eine Homotopie $H : f \sim g$ gilt:
$$\mathrm{tr}_{g \mapsto f \sim g}(p)(H) = (x : A) \mapsto H_x \cdot \mathrm{hap}(p)_x$$
Sei $(f, x \mapsto \mathrm{refl}_{f(x)})$ unser Kontraktionszentrum. Dann ist für jedes weitere $(g, H) : \sum_{g:\prod_{x:A} B(x)} f \sim g$ durch die Konstruktion am Anfang des Beweises eine Gleichheit $p_H : f = g$ gegeben. Um zu zeigen, dass $(f, x \mapsto \mathrm{refl}_{f(x)}) = (g, H)$ gilt, müssen wir nach der Charakterisierung in Lemma 2.5.1 also nur noch zeigen:
$$\mathrm{tr}_{g \mapsto f \sim g}(p_H)(x \mapsto \mathrm{refl}_{f(x)}) = H$$
Linke und rechte Seite dieser Gleichung sind abhängige Funktionen des Typs $\prod_{x:A} f(x) = g(x)$, also können wir die Konstruktion "p_H" anwenden, um die benötigte Gleichheit aus einer Punktweisen Gleichheit, also aus
$$\mathrm{tr}_{g \mapsto f \sim g}(p_H)(x \mapsto \mathrm{refl}_{f(x)}) \sim H$$
zu folgern. Diese gilt aber bereits nach der Berechnung des Transports und (a). □

Nachdem wir nun die vermutlich einfachsten nicht-trivialen höheren induktiven Typen (HIT) kennen gelernt haben, wollen wir uns zum Abschluss des Abschnitts noch einem etwas komplizierteren HIT widmen, dem sogenannten Mengenquotient.

Regeln 3.1.11
Seien $A : \mathcal{U}$ und $R : A \to A \to \mathcal{U}$. Dann ist $A/R : \mathcal{U}$ der höhere induktive Typ mit den folgenden Konstruktoren:
$$\begin{aligned}
[_] &: A \to A/R \\
\mathrm{eq}/ &: \prod_{x,y:A} R(x, y) \to [x] =_{A/R} [y] \\
\mathrm{set}/ &: \prod_{x,y:A/R} \prod_{p,q:x=y} p = q
\end{aligned}$$
Der Typ A/R heißt *Mengenquotient.*

Bemerkung 3.1.12
A/R ist stets eine Menge. Und es gibt folgendes Rekursionsprinzip: Für jede Menge B, Funktion $f : A \to B$ und Nachweis, dass die Relation respektiert wird, also einen Term in

$$r : \prod_{x,y:A} R(x,y) \to f(x) =_B f(y)$$

gibt es eine Funktion

$$\mathrm{rec}_{A/R}(B, f, r) : A/R \to B$$

Analog gibt es ein vereinfachtes Induktionsprinzip. Für jede Aussage $P : A/R \to \mathcal{U}$ (also $P(x)$ Aussage für jedes $x : A/R$) reicht es $P([a])$ für alle $a : A$ zu zeigen, um $\prod_{x:A/R} P(x)$ zu zeigen.

Beweis Auf die Fälle für die höheren Konstruktoren kann jeweils verzichtet werden, da die entpsrechenden Gleichheiten in Mengen bzw. Aussagen stets existieren. □

Wir werden nun die Konstruktion von $\mathbb{Z}$ als Quotient der Menge der Paare in $\mathbb{N} \times \mathbb{N}$ mithilfe von Mengenquotienten nachbauen. Die Paare $(n, k) : \mathbb{N} \times \mathbb{N}$ darf man sich als Differenz "$n - k$" vorstellen, was auch die folgende Relation erklärt.

Definition 3.1.13
Sei für $(n, k), (n', k') : \mathbb{N} \times \mathbb{N}$:

$$(n, k) \sim_{\mathbb{Z}} (n', k') :\equiv n + k' =_{\mathbb{N}} n' + k$$

Die *Ganzen Zahlen* sind wie folgt als Mengenquotient gegeben:

$$\mathbb{Z} :\equiv \mathbb{N} \times \mathbb{N}/\sim_{\mathbb{Z}}$$

Definition 3.1.14
Die Funktion $\mathrm{succ}_{\mathbb{Z}} : \mathbb{Z} \to \mathbb{Z}$ sei durch Rekursion gegeben durch:

$$s(n, k) :\equiv (\mathrm{succ}_{\mathbb{N}}(n), k)$$

und den Nachweis der Wohldefiniertheit: Sei $(n, k) \sim_{\mathbb{Z}} (n', k')$, dann gilt $n + k' = n' + k$, also auch $\mathrm{succ}_{\mathbb{N}}(n + k') = \mathrm{succ}_{\mathbb{N}}(n' + k)$. Letzteres ist nach der Definition der Addition $\mathrm{succ}_{\mathbb{N}}(n) + k' = \mathrm{succ}_{\mathbb{N}}(n') + k$, also gilt $s(n, k) \sim_{\mathbb{Z}} s(n', k')$.

3.2 n-Typen

In Definition 1.6.5 hatten wir bereits kontrahierbare Typen, Aussagen und Mengen definiert. Die Mengen haben den alternativen Namen 0-Typen und zu Aussagen, kann man auch -1-Typen und zu kontrahierbaren Typen -2-Typen sagen. Wir werden in diesem Abschnitt n-Typen definieren, für $n = -2, -1, 0, 1, \ldots$. Dazu sei zunächst:

Definition 3.2.1
Es sei $\mathbb{N}_{-2}$ der induktive Typ mit Konstruktoren $-2 : \mathbb{N}_{-2}$ und $\mathrm{succ}_{\mathbb{N}_{-2}} : \mathbb{N}_{-2} \to \mathbb{N}_{-2}$. Für $\mathrm{succ}_{\mathbb{N}_{-2}}(n)$ schreiben wir $n + 1$. Die Elemente von $\mathbb{N}_{-2}$ nennen wir auch *Abschneidungslevel*.

Damit können wir n-Typen für jeden Abschneidungslevel n definieren:

Definition 3.2.2
(a) Für jeden Typ A sei der Typ is-n-type wie folgt per Rekursion über $n : \mathbb{N}_{-2}$ definiert:

$$\text{is-}(-2)\text{-type}(A) :\equiv \mathrm{isContr}(A)$$
$$\text{is-}(n+1)\text{-type}(A) :\equiv \prod_{x,y:A} \text{is-}n\text{-type}(x =_A y)$$

(b) Der *Typ der n-Typen* ist für $n : \mathbb{N}_{-2}$:

$$n\text{−Type} :\equiv \sum_{A:\mathcal{U}} \text{is-}n\text{-type}(A)$$

Diesen Typ gibt es also für jeden Universenlevel.

Lemma 3.2.3
Sind Typen $A, B : \mathcal{U}$ äquivalent und ist A ein n-Typ, dann ist auch B ein n-Typ.

Beweis Univalenz. □

Lemma 3.2.4
Sei $A : \mathcal{U}$ und $B : A \to \mathcal{U}$ so, dass $B(x)$ für jedes $x : A$ ein n-Typ ist. Dann ist $\prod_{x:A} B(x)$ ein n-Typ.

Beweis Wir beweisen zunächst den Fall $n \equiv -2$. Seien also die $B(x)$ kontrahierbar. Dann gibt es eine abhängige Funktion $z : \prod_{x:A} B(x)$, die jedes $x : A$ auf das Kontraktionszentrum von $B(x)$ abbildet. Es reicht also zu zeigen, dass $\prod_{x:A} B(x)$ eine Aussage ist. Seien dazu $f, g : \prod_{x:A} B(x)$. Wir wollen $f = g$ zeigen. Mit abhängiger Funktionsextensionalität reicht es zu zeigen, dass $f \sim g$ gilt. Wir wissen aber, dass für jedes $x : A$ bereits $f(x) = g(x)$ gilt, da $B(x)$ eine Aussage ist.

Wir zeigen nun die eigentliche Aussage per Induktion über den Abschneidungslevel. Seien nun also die $B(x)$ $n+1$-Typen. Für $f, g : \prod_{x:A} B(x)$ gilt $f = g \simeq f \sim g \equiv \prod_{x:A} f(x) =_{B(x)} g(x)$. Letzteres ist ein Gleichheitstyp in einem $n+1$-Typ, also ein n-Typ. Damit können wir die Induktionshypothese anwenden und sehen, dass auch $f = g$ ein n-Typ sein muss und damit dass $\prod_{x:A} B(x)$ ein $n+1$-Typ ist. □

Lemma 3.2.5
Sei $A : \mathcal{U}$. Für jedes $n : \mathbb{N}_{-2}$ gilt isProp(is-n-type(A)).

Beweis Induktion über den Abschneidungslevel. Wir zeigen also zunächst: isProp(isContr(A)). Seien also

$$(a, H), (a', H') : \text{isContr}(A) \equiv \sum_{x:A} \prod_{y:A} x = y$$

Mit Lemma 2.5.1 müssen wir also zeigen:

$$\sum_{p:a=a'} \text{tr}_{x \mapsto \prod_{y:A} x=y}(p)(H) = H'$$

wobei der Transport die Linkskonkatenation mit p^{-1} ist. Wir wählen $p :\equiv H_{a'}$. Dann ist noch zu zeigen:

$$\prod_{y:A} H_{a'}^{-1} \cdot H_y = H'_y$$

Das lässt sich etwas überraschend auf dem Umweg über die Verallgemeinerung

$$\prod_{q:x=y} H_x^{-1} \cdot H_y = q$$

mit Gleichheitsinduktion zeigen. Damit ist also isContr(A) eine Aussage.

Jetzt nehmen wir also an, dass is-n-type(A) eine Aussage ist und zeigen, dass is-$(n+1)$-type(A) eine Aussage ist. Nach Definition gilt

$$\text{is-}(n+1)\text{-type}(A) \equiv \prod_{x,y:A} \text{is-}n\text{-type}(x =_A y)$$

Für festes $x : A$ ist also mit Lemma 3.2.4 der Typ $\prod_{y:A}$ is-n-type($x =_A y$) eine Aussage. Mit erneutem Anwenden von Lemma 3.2.4, ist also auch $\prod_{x:A} \prod_{y:A}$ is-n-type($x =_A y$) eine Aussage. □

Damit können wir sofort einsehen:

Korollar 3.2.6
Seien $A, B : \mathcal{U}$ und $f : A \to B$. Die folgende Definition von isEquiv(f) ist eine Aussage:

$$\prod_{y:B} \text{isContr}(f^{-1}(y))$$

Beweis Zunächst wissen wir durch Lemma 3.2.5, dass isContr($f^{-1}(y)$) für jedes $y : B$ eine Aussage ist. Mit Lemma 3.2.4 wissen wir außerdem, dass das abhängige Produkt über Aussagen wieder eine Aussage ist. □

Wir haben in Lemma 3.2.4 bereits gesehen, dass die Gesamtheit der n-Typen für festes n unter abhängigen Produkten abgeschlossen ist. Nun werden wir sehen, dass Ähnliches für den Großteil der Konstruktionen gilt, die wir soweit kennen.

Bemerkung 3.2.7

(a) Jeder n-Typ ist auch ein $(n+1)$-Typ.

(b) Wenn A ein n-Typ ist, dann sind auch alle Gleichheitstypen $x =_A y$ für $x, y : A$ wieder n-Typen.

Beweis (a) Sei zunächst $n \equiv -2$, also A kontrahierbar mit Zeugen $(z, H) : \mathrm{isContr}(A)$. Für $x, y : A$ wählen wir $H_x^{-1} \cdot H_y : x =_A y$ als Kontraktionszentrum. Per Gleichheitsinduktion gibt es für beliebiges $p : x = y$ eine Gleichheit $H_x^{-1} \cdot H_y = p$.

Für den Induktionsschritt sei nun A ein $(n+1)$-Typ. Damit sind alle $x =_A y$ n-Typen. Nach Induktionshypothese ist $x = y$ also auch ein $(n+1)$-Typ und damit A ein $(n+2)$-Typ.

(b) Für $n \equiv -2$ haben wir das bereits in (a) nachgerechnet. Nach Definition ist für einen $n+1$-Typ A jeder Gleichheitstyp ein n-Typ, also nach (a) auch ein $n+1$-Typ. □

Wenn man bedenkt, dass es für jeden Typ A eine Äquivalenz $A \simeq \sum_{x:A} \mathbf{1}$ gibt, wird klar, dass für abhängige Summen kein direktes Analogon zu Lemma 3.2.4 gelten kann. Weiter wäre das seltsam, weil der Spezialfall von Produkten symmetrisch ist. Es liegt also Nahe, für abhängige Summen auch Bedingungen an die Basis zu stellen, um den Abschneidungslevel zu erhalten:

Lemma 3.2.8
Seien $A : \mathcal{U}$ und $B : A \to \mathcal{U}$. Wenn A ein n-Typ und $B(x)$ für jedes $x : A$ ein n-Typ ist, dann ist auch die abhängige Summe $\sum_{x:A} B(x)$ ein n-Typ.

Beweis Sei $n \equiv -2$, also A kontrahierbar und für jedes $x : A$ $B(x)$ kontrahierbar. Wir wählen $(z, b) : \sum_{x:A} B(x)$ als Kontraktionszentrum, für ein Kontraktionszentrum $z : A$ von A und $b : B(z)$ von $B(z)$. Seien $z' : A$ und $b' : B(z')$. Es gilt:

$$(z, b) = (z', b') \simeq \sum_{p:z=z'} \mathrm{tr}_B(p)(b) = b'$$

Die beiden Gleichheiten auf der rechten Seite gibt es jeweils wegen Kontrahierbarkeit.

Seien nun A und jedes $B(x)$ ein $(n+1)$-Typ. Dann ist ohne Einschränkung zu zeigen, dass für $x, y : A$ und $b : B(x)$, $b' : B(y)$ der Typ

$$(x, b) = (y, b') \simeq \sum_{p:x=y} \mathrm{tr}_B(p)(b) = b'$$

ein n-Typ ist. Die rechte Seite ist aber eine abhängige Summe von n-Typen, also nach Induktionshypothese ein n-Typ. □

Bemerkung 3.2.9
Sei $n \geq 0$. Koprodukte von n-Typen sind n-Typen.

Beweis Seien $A, B : \mathcal{U}$ und $(n+1) \geq -1$ ein Abschneidungslevel. Wir zeigen, dass $A \sqcup B$ ein $(n+1)$-Typ ist, indem wir zeigen, dass alle Gleichheitstypen von $A \sqcup B$ n-Typen sind. Nach Übungsaufgabe 3 von Blatt 7, ist der Gleichheitsyp des Koprodukts $A \sqcup B$ faserweise äquivalent zu:

$$\begin{aligned}
\mathrm{Eq}_{\sqcup}(\iota_1(a), \iota_1(a')) &:\equiv (a =_A a') \\
\mathrm{Eq}_{\sqcup}(\iota_1(a), \iota_2(b)) &:\equiv \emptyset \\
\mathrm{Eq}_{\sqcup}(\iota_2(b), \iota_1(a)) &:\equiv \emptyset \\
\mathrm{Eq}_{\sqcup}(\iota_2(b), \iota_2(b')) &:\equiv (b =_B b')
\end{aligned}$$

Der Typ $\emptyset$ ist eine Aussage. Da $n \geq -1$ ist, ist also $\emptyset$ mit Bemerkung 3.2.7 auch ein n-Typ. Die Typen $a = a'$ und $b = b'$ sind auch jeweils n-Typen, also sind alle Gleichheitstypen von $A \sqcup B$ n-Typen. □

Bemerkung 3.2.10

(a) Seien $A, B : \mathcal{U}$ und $f : A \to B$. Die Fasern von $\mathrm{ap}(f, \_) : x = y \to f(x) = f(y)$ sind genau dann n-Typen, wenn jede Faser von f ein $(n+1)$-Typ ist.

(b) Untertypen von n-Typen sind n-Typen.

Beweis (a) Die Aussage folgt, wenn die Gleichheitstypen jeder Faser von f äquivalent zu Fasern von $\mathrm{ap}(f, \_)$ sind und umgekehrt. Seien also $y : B$ und $(x, p), (x', p') : f^{-1}(y)$, dann gilt:

$$\begin{aligned}(x,p) = (x',p') &\simeq \sum_{q:x=x'} \mathrm{tr}_{f(\_)=y}(q)(p) = p' \\ &\simeq \sum_{q:x=x'} f(q)^{-1}\cdot p = p' \\ &\simeq \sum_{q:x=x'} f(q) = p'^{-1}\cdot p \\ &\simeq \mathrm{ap}(f,\_)^{-1}(p'^{-1}\cdot p)\end{aligned}$$

Und für $p : f(x) = f(y)$ gilt:

$$\begin{aligned}\mathrm{ap}(f,\_)^{-1}(p) \sum_{q:x=y} f(q) &= p \\ &\simeq \sum_{q:y=x} f(q)^{-1}\cdot p = \mathrm{refl}_{f(x)} \\ &\simeq ((y,p) =_{f^{-1}(f(x))} (x, \mathrm{refl}_{f(x)}))\end{aligned}$$

(b) Das folgt aus (a) für $n \equiv -1$ und Übungsaufgabe 4 von Blatt 9. □

Es ist nicht der Fall, dass das Universum der n-Typen, also der Typ

$$n\text{-Type} \equiv \sum_{A:\mathcal{U}} \text{is-}n\text{-type}(A)$$

wieder ein n-Typ ist. Ein Gegenbeispiel ist durch $(\mathbf{2} = \mathbf{2}) \simeq \mathbf{2}$ gegeben: $\mathbf{2}$ ist ein 0-Typ, aber es gibt zwei verschiedene Gleichheiten zwischen $\mathbf{2}$ und $\mathbf{2}$ im Typ 0-Type. Letzterer könnte also bestenfalls noch ein 1-Typ sein. Das ist tatsächlich so:

Bemerkung 3.2.11
Der Typ der n-Typen ist ein $(n+1)$-Typ.

Beweis Für $n \equiv -2$, sind alle Typen äquivalent zu $\mathbf{1}$ und $(\mathbf{1} = \mathbf{1}) \simeq \mathbf{1}$ ist ein (-2)-Typ, also auch ein (-1)-Typ.

Seien nun $A, B : \mathcal{U}$ zwei (n+1)-Typen. Wir wollen zeigen, dass im Typ der (n+1)-Typen also für Paare $(A, X), (B, Y) : (n+1)$-Type gilt, dass $(A, X) = (B, Y)$ ein n-Typ ist. Da X, Y nur Elemente von Aussagen sind gilt:

$$\big((A,X) =_{(n+1)\text{-Type}} (B,Y)\big) \simeq (A =_{\mathcal{U}} B) \simeq (A \simeq B)$$

Letzteres ist ein Untertyp von $A \to B$, was nach Lemma 3.2.4 ein n-Typ ist. □

3.3 Überlagerungen und $\Omega(S^1)$

Bevor wir uns dem eigentlichen Thema zuwenden, brauchen wir noch eine Variante der ganzen Zahlen mit einem Rekursionsprinzip, das uns auch Abbildungen in nicht notwendigerweise 0-abgeschnittene Typen erlaubt.

Regeln 3.3.1
Zunächst sei $\mathbb{N}_1 : \mathcal{U}$ der induktive Typ mit Konstruktoren $1_{\mathbb{N}_1} : \mathbb{N}_1$ und $\mathrm{succ}_{\mathbb{N}_1} : \mathbb{N}_1 \to \mathbb{N}_1$. Nun sei $\mathbb{Z}' : \mathcal{U}$ der induktive Typ mit Konstruktoren:

$$\begin{aligned}0_{\mathbb{Z}'} &: \mathbb{Z}' \\ \mathrm{pos} &: \mathbb{N}_1 \to \mathbb{Z}' \\ \mathrm{neg} &: \mathbb{N}_1 \to \mathbb{Z}'\end{aligned}$$

Wir nennen den Typ $\mathbb{Z}'$ die *(induktiven) ganzen Zahlen* und werden auch später auf den Strich verzichten und einfach $\mathbb{Z}$ schreiben. Als Induktionsprinzip ergibt sich also, dass wir Konstruktionen für die positiven Zahlen, die negativen Zahlen und die 0 ausführen müssen.

Bemerkung 3.3.2
Die Typen $\mathbb{N}_1$ und $\mathbb{Z}'$ sind Mengen.

Beweis Nach Theorem 2.4.2 ist der Typ $\mathbb{N}$ eine Menge und dieser ist äquivalent zu $\mathbb{N}_1$. Für $\mathbb{Z}'$ führen wir einen Beweis mit der üblichen Methode. Sei dazu $\mathrm{Eq}_{\mathbb{Z}'} : \mathbb{Z}' \to \mathbb{Z}' \to \mathcal{U}$ gegeben durch:

$$\begin{aligned}\mathrm{Eq}_{\mathbb{Z}'}(0_{\mathbb{Z}'}, 0_{\mathbb{Z}'}) &:\equiv \mathbf{1}\\ \mathrm{Eq}_{\mathbb{Z}'}(\mathrm{pos}(n), \mathrm{pos}(k)) &:\equiv n =_{\mathbb{N}_1} k\\ \mathrm{Eq}_{\mathbb{Z}'}(\mathrm{neg}(n), \mathrm{neg}(k)) &:\equiv n =_{\mathbb{N}_1} k\\ \text{übrige Fälle} &:\equiv \emptyset\end{aligned}$$

Den Reflexivitätsterm $r_{\mathbb{Z}'} : \prod_{x:\mathbb{Z}'} \mathrm{Eq}_{\mathbb{Z}'}(x,x)$ können wir definieren als $*$, im Fall $x \equiv 0$ und als refl_n in den Fällen $x \equiv \mathrm{pos}(n)$ und $x \equiv \mathrm{neg}(n)$. Nun ist etwa für $x \equiv \mathrm{pos}(n)$ zu zeigen, dass

$$\sum_{x:\mathbb{Z}'} \mathrm{Eq}_{\mathbb{Z}'}(\mathrm{pos}(n), x)$$

kontrahierbar ist. Per Induktion über x ist das nur noch für $x \equiv \mathrm{pos}(k)$ zu zeigen, da in allen anderen Fällen nur etwas für alle Elemente des leeren Typs zu zeigen ist. Im Fall $x \equiv \mathrm{pos}(k)$ müssen wir allerdings nur zeigen, dass ein Element $(k,p) : \sum_{k:\mathbb{N}_1} n = k$ gleich (n, refl) ist - das können wir mit Lemma 1.6.9 zeigen. □

Bemerkung 3.3.3
Es gilt $\mathbb{Z} \simeq \mathbb{Z}'$. Wir schreiben daher ab jetzt $\mathbb{Z}$ für beide.

Beweis Übungsaufgabe. □

Nun können wir uns den Überlagerungen zuwenden. Speziell werden wir mit der universellen Überlagerung des Kreises beginnen. In der Topologie wird diese als Kopie der reellen Gerade $\mathbb{R}$ konstruiert, die als Helix über dem topologischen Kreis $\mathbb{S}^1 = \{(x,y) \in \mathbb{R}^2 | x^2 + y^2 = 1\}$ liegt. Als konkrete Projektion von $\mathbb{R}$ auf $\mathbb{S}^1$ kann etwa

$$(t : \mathbb{R}) \mapsto (\cos(2\pi t), \sin(2\pi t))$$

gewählt werden. Unser folgender abstrakter Nachbau dieser Situation handelt zwar nur von den Homotopietypen der beteiligten topologischen Räume, ist aber trotzdem überraschend ähnlich. So werden etwa die Fasern der abstrakten Projektion ebenfalls $\mathbb{Z}$ sein. Tatsächlich werden wir sogar genau damit beginnen.

Unser Ziel wird zunächst sein, zu beweisen, dass der Gleichheitstyp $* =_{S^1} *$ äquivalent zu $\mathbb{Z}$ ist. Dazu geben wir diesen spezielleren Gleichheitstypen noch ihren üblichen Namen.

Definition 3.3.4
(a) Ein *punktierter Typ* ist ein Paar $(A,a) : \sum_{A:\mathcal{U}} A$. Wir werden oft von punktierten Typen $A : \mathcal{U}$ sprechen und implizit den Punkt mit $* : A$ bezeichnen.

(b) Der *Schleifenraum* eines punktierten Typs $(A,*)$ ist der Typ $\Omega(A,*) :\equiv (* =_A *)$.

Definition 3.3.5
Die *universelle Überlagerung* von S^1 ist der abhängige Typ $\tilde{S^1} : S^1 \to \mathcal{U}$ gegeben durch

$$\tilde{S^1} :\equiv \mathrm{rec}_{S^1}(\mathcal{U}, \mathbb{Z}, \mathrm{ua}(\mathrm{succ}_{\mathbb{Z}}))$$

Es gilt also $\tilde{S^1}(*) = \mathbb{Z}$ und $\mathrm{ap}(\tilde{S^1}, l) = \mathrm{ua}(\mathrm{succ}_{\mathbb{Z}})$.

Die letztere Gleichheit lässt sich etwas greifbarer machen und entspricht im topologischen Fall der Monodromieaktion eines Erzeugers der Fundamentalgruppe.

Bemerkung 3.3.6
Es gilt $\mathrm{tr}_{\tilde{S^1}}(l) =_{\mathbb{Z}\to\mathbb{Z}} \mathrm{succ}_{\mathbb{Z}}$.

Beweis Allgemein gilt für einen abhängigen Typ $B : A \to \mathcal{U}$:

$$\prod_{x,y:A} \prod_{p:x=y} \mathrm{ua}(\mathrm{tr}_B(p)) = \mathrm{ap}(B,p)$$

was sich per Induktion über p beweisen lässt, da $\mathrm{ua}(\mathrm{id}_{B(x)}) = \mathrm{refl}_{B(x)}$ gilt. Damit gilt also

$$\mathrm{ua}(\mathrm{tr}_{\tilde{S^1}}(l)) = \mathrm{ap}(\tilde{S^1}, l) = \mathrm{ua}(\mathrm{succ}_{\mathbb{Z}})$$

womit die Behauptung folgt, weil ua eine Äquivalenz ist. □

Das bedeutet, dass wir den Transport in $\tilde{S^1}$ benutzen können, um Gleichheiten der Form $* =_{S^1} *$, also Elemente des Schleifenraums $\Omega(S^1, *)$, mit l zu vergleichen. Da l einmal den Kreis durchläuft, kann man mit der folgenden Definition messen, wie oft eine Gleichheit in $\Omega(S^1, *)$ den Kreis S^1 durchläuft.

Definition 3.3.7
Die *Windungszahl* einer Gleichheit in $\Omega(S^1, *)$ ist der Wert unter der Abbildung

$$\begin{aligned} w : \Omega(S^1, *) &\to \mathbb{Z} \\ w(p) &:\equiv \mathrm{tr}_{\tilde{S^1}}(p)(0) \end{aligned}$$

Die Windungszahl wird sich später als Äquivalenz herausstellen und hat die Eigenschaften eines Gruppenhomomorphismus. Wir zeigen das zunächst nur für Addition mit 1.

Bemerkung 3.3.8
Für alle $p : \Omega(S^1, *)$ gilt $w(p \cdot l) = \mathrm{succ}_{\mathbb{Z}}(w(p))$.

Beweis Das ist durch die Verträglichkeit von Konkatenation und Komposition von Transporten gegeben:

$$\begin{aligned} w(p \cdot l) &= \mathrm{tr}_{\tilde{S^1}}(p \cdot l)(0) \\ &= (\mathrm{tr}_{\tilde{S^1}}(l) \circ \mathrm{tr}_{\tilde{S^1}}(p))(0) \\ &= \mathrm{succ}_{\mathbb{Z}}(\mathrm{tr}_{\tilde{S^1}}(p)(0)) \\ &= \mathrm{succ}_{\mathbb{Z}}(w(p)) \end{aligned}$$

Wenn die Windungszahl eine Äquivalenz ist, bedeutet das, dass jedes $p : \Omega(S^1, *)$ von der Form l^k für $k : \mathbb{Z}$ ist. Also sollte die Inverse von w durch wiederholte Konkatenation von $l : \Omega(S^1, *)$ gegeben sein.

Definition 3.3.9
Die *Potenzfunktion* $l^{(\_)} : \mathbb{Z} \to \Omega(S^1, *)$ ist $\mathbb{Z}'$- und $\mathbb{N}_1$-rekursiv gegeben durch

$$\begin{aligned} l^0 &:\equiv \mathrm{refl}_* \\ l^{\mathrm{pos}(1)} &:\equiv l \\ l^{\mathrm{pos}(n+1)} &:\equiv l^{\mathrm{pos}(n)} \cdot l \\ l^{\mathrm{neg}(1)} &:\equiv l^{-1} \\ l^{\mathrm{neg}(n+1)} &:\equiv l^{\mathrm{neg}(n)} \cdot l^{-1} \end{aligned}$$

Anstatt direkt zu zeigen, dass $l^{(\_)}$ und w zueinander invers sind, werden wir eine faserweise Äquivalenz der Form

$$\prod_{x:S^1} * =_{\tilde{S^1}} x \to \tilde{S^1}(x)$$

konstruieren.

Lemma 3.3.10
Es gibt eine faserweise Abbildung

$$W : \prod_{x:S^1} * =_{\tilde{S^1}} x \to \tilde{S^1}(x)$$

mit:

$$W(*) :\equiv w : * = * \to \mathbb{Z}$$

Beweis Um einzusehen, dass W durch S^1-Induktion definiert werden kann, müssen wir berechnen, wie die Abbildung w entlang von $l : * = *$ transportiert wird. Dazu überlegen wir zunächst, was der Transport in einer abstrakteren Situation wäre, nämlich in einem abhängigen Typ der Form:

$$(x : A) \mapsto B(x) \to C(x)$$

für zwei abhängige Typen $B : A \to \mathcal{U}$ und $C : A \to \mathcal{U}$. Die naheliegende Vermutung ist, dass wir mit den Transporten in B und C prä- und postkomponieren müssen. Es kann mit Induktion über p geklärt werden, dass gilt:

$$\mathrm{tr}_{(x:A)\mapsto B(x)\to C(x)}(p)(f) = \mathrm{tr}_C(p) \circ f \circ \mathrm{tr}_B(p^{-1})$$

Den Transport in $x \mapsto * = x$ kennen wir, das ist Rechtskonkatenation. Damit die induktive Definition von W wie behauptet funktioniert, müssen wir also verifizieren, dass gilt:

$$\mathrm{tr}_{\tilde{S^1}}(l) \circ w \circ \mathrm{tr}_{*=\_}(l^{-1}) = w \qquad \square$$

Das lässt sich wie folgt punktweise für $p : * = *$ berechnen:

$$\begin{aligned}
\mathrm{tr}_{\tilde{S^1}}(l) \circ w \circ \mathrm{tr}_{*=_}(l^{-1})(p) &= \mathrm{tr}_{\tilde{S^1}}(l) \circ w(p \cdot l^{-1}) \\
&= \mathrm{tr}_{\tilde{S^1}}(l) \circ \mathrm{tr}_{\tilde{S^1}}(p \cdot l^{-1})(0) \\
&= \mathrm{tr}_{\tilde{S^1}}(l) \circ \mathrm{tr}_{\tilde{S^1}}(l^{-1}) \circ \mathrm{tr}_{\tilde{S^1}}(p)(0) \\
&= \mathrm{tr}_{\tilde{S^1}}(p)(0) \\
&= w(p)
\end{aligned}$$

Lemma 3.3.11
Es gibt auch eine Fortsetzung der Potenzfunktion $k \mapsto l^k : \mathbb{Z} \to \Omega(S^1, *)$, also eine faserweise Abbildung

$$P : \prod_{x:S^1} \tilde{S^1}(x) \to * = x$$

mit $P_* = k \mapsto l^k$

Beweis Um P induktiv definieren zu können, müssen wir zeigen:

$$\mathrm{tr}_{(x:S^1) \mapsto \tilde{S^1}(x) \to * = x}(l)(k \mapsto l^k) = (k \mapsto l^k) \qquad \square$$

Analog zum beweis von Lemma 3.3.10 ergibt sich:

$$\begin{aligned}
\mathrm{tr}_{(x:S^1) \mapsto \tilde{S^1}(x) \to * = x}(l)(k \mapsto l^k) &= \mathrm{tr}_{*=_}(l) \circ (k \mapsto l^k) \circ \mathrm{tr}_{\tilde{S^1}}(l^{-1}) \\
&= (p \mapsto p \cdot l) \circ (k \mapsto l^k) \circ \mathrm{tr}_{\tilde{S^1}}(l^{-1}) \\
&= (k \mapsto l^{\mathrm{succ}_{\mathbb{Z}}(k)}) \circ \mathrm{succ}_{\mathbb{Z}}^{-1} \\
&= (k \mapsto l^k)
\end{aligned}$$

Theorem 3.3.12
Es gilt:

(a) W und P sind faserweise zueinander invers.

(b) $w : \Omega(S^1, *) \to \mathbb{Z}$ ist eine Äquivalenz mit Inversem $k \mapsto l^k : \mathbb{Z} \to \Omega(S^1, *)$.

Beweis (a) Wir rechnen punktweise nach. Sei $x : S^1$, dann soll für alle $p : * = x$ gelten:

$$P_x(W_x(p)) = p$$

Dank Induktion mit Basispunkt müssen wir das nur für $p \equiv \mathrm{refl}$ berechnen:

$$P_*(W_*(\mathrm{refl})) = P_*(0) = \mathrm{refl}$$

Für die andere Richtung müssen wir zeigen, dass für alle $x : S^1$ und $k : \tilde{S^1}(x)$ gilt:

$$W_x(P_x(y)) = k$$

Das wollen wir mit Kreisinduktion zeigen. Zunächst berechnen wir also:

$$W_*(P_*(k)) = W_*(l^k)$$

Wir sind mit dem Induktionsangfang fertig, wenn $W_*(l^k) \equiv w(l^k) = k$ gilt. Das können wir mit $\mathbb{Z}'$-Induktion und $\mathbb{N}_1$-Induktion zeigen:

$$\begin{aligned}
w(l^0) &= \mathrm{tr}_{\tilde{S^1}}(\mathrm{refl})(0) = 0 \\
w(l^{\mathrm{pos}(1)}) &= w(l) = \mathrm{succ}_{\mathbb{Z}}(0) = \mathrm{pos}(1) \\
w(l^{\mathrm{pos}(n+1)}) &= w(l^{\mathrm{pos}(n)} \cdot l) = \mathrm{succ}_{\mathbb{Z}}(w(l^{\mathrm{pos}(n)})) = \mathrm{pos}(n+1) \\
w(l^{\mathrm{neg}(1)}) &= w(l^{-1}) = \mathrm{succ}^{-1}(0) = \mathrm{neg}(1) \\
w(l^{\mathrm{neg}(n+1)}) &= w(l^{\mathrm{neg}(n)} \cdot l^{-1}) = \mathrm{succ}^{-1}(w(l^{\mathrm{neg}(n)})) = \mathrm{neg}(n+1)
\end{aligned}$$

Nun ist noch zu zeigen, dass Transport der soeben konstruierten Gleichheit entlang $l : * = *$, wieder diese Gleichheit ist. Was auch immer wir genau konstruieren müssen, es ist auf jeden Fall eine Gleichheit zwischen Gleichheiten im Typ $\mathbb{Z}$. Da letzterer Typ eine Menge ist, existieren also alle Gleichheiten dieser Art.

(b) In (a) für $x : S^1$ den Punkt $*$ einsetzen. □

Die Eigenschaft der abhängigen Typen $\tilde{S^1}$ und $(x : S^1) \mapsto * = x$, dass alle Werte Mengen sind, ist bereits die definierende Eigenschaft von Überlagerungen. Allerdings werden wir die Projektionen dieser abhängigen Typen Überlagerungen nennen. Dabei sollte man bedenken, dass es sich hierbei nur um die Homotopietypen von Überlagerungen handelt.

Definition 3.3.13
Sei $A : \mathcal{U}$ ein Typ.

(a) Ein abhängiger Typ $B : A \to \mathcal{U}$ heißt n-abgeschnitten, wenn jedes $B(x)$ ein n-Typ, also n-abgeschnitten ist.

(b) Eine Abbildung heißt n-abgeschnitten, wenn alle ihre Fasern n-abgeschnitten sind.

(c) Eine Abbildung $f : A \to B$ heißt *Überlagerung*, wenn sie 0-abgeschnitten ist.

3.4 Sphären, Homotopiegruppen und n-Abschneidungen

Definition 3.4.1
Seien $(A, *_A)$ und $(B, *_B)$ punktierte Typen.

(a) Eine *punktierte Abbildung* oder *Abbildung punktierter Typen* ist eine Funktion $f : A \to B$ zusammen mit einer Gleichheit $p_f : f(*_A) = *_B$. Der Typ der punktierten Abbildungen ist:

$$(A, *_A) \to^* (B, *_B) :\equiv \sum_{f:A\to B} f(*_A) = *_B$$

Wir werden auch $A \to^* B$ schreiben, wenn die Punktierung klar ist und etwa $(f, p_f) : (A, *_A) \to^* (B, *_B)$, um der Gleichheit der punktierten Abbildung einen Namen zu geben.

(b) Der *Typ der punktierten Typen* ist

$$\mathcal{U}^* :\equiv \sum_{A:\mathcal{U}} A$$

(c) Den Schleifenraum $\Omega(A, *_A)$ werden wir nun auch als punktierten Raum mit dem Punkt refl_{*_A} auffassen.

(d) Für ein punktierte Abbildung $(f, p_f) : (A, *_A) \to^* (B, *_B)$ ist

$$\Omega(f, p_f) :\equiv (p : *_A = *_A) \mapsto p_f^{-1} \cdot f(p) \cdot p_f$$

(e) Die Typen $\mathbf{1}, \mathbf{2}, \mathbb{N}$ und $\mathbb{Z}$ sind jeweils durch 0 punktiert. Alle Summen $\sum_{x:A} B(x)$ mit punktiertem A und $B(*)$ sind punktiert und abhängige Produkte $\prod_{x:A} B(x)$ mit punktierten $B(x)$ sind punktiert.

Definition 3.4.2 (n-facher Schleifenraum)
Für $n : \mathbb{N}$ und $(A, *_A)$ sei der *n-fache Schleifenraum* $\Omega^n(A, *_A)$ rekursiv gegeben durch:

$$\Omega^0 :\equiv (A, *_A)$$
$$\Omega^{n+1} :\equiv \Omega(\Omega^n(A, *_A))$$

wobei wir Schleifenräume stets als punktiert auffassen.

Lemma 3.4.3
Sei $A : \mathcal{U}$, dann sind für $n \geq -1$ äquivalent:

i) A ist ein $(n+1)$-Typ.

ii) Der Schleifenraum $\Omega(A, x)$ ist ein n-Typ für jedes $x : A$.

Beweis (Idee) Jeder Gleichheitstyp $x =_A y$ ist äquivalent zum Schleifenraum $\Omega(A, x)$, wenn es ein $p : x =_A y$ gibt. Tatsächlich reicht diese Aussage schon, denn für $n \geq -1$ und jedes X gilt:

$$(X \to \text{is-}n\text{-type}(X)) \to \text{is-}n\text{-type}(X) \qquad \square$$

Definition 3.4.4
Seien $A, B, C : \mathcal{U}$ und $f : A \to C$, $g : B \to C$. Es ist also ein Winkel gegeben:

$$\begin{array}{ccc} & & B \\ & & \downarrow{\scriptstyle g} \\ A & \xrightarrow[f]{} & C \end{array}$$

(a) Sei $X : \mathcal{U}$. Ein *X-Kegel* besteht aus Abbildungen $\varphi : X \to A$ und $\psi : X \to B$ zusammen mit einer Homotopie $H : f \circ \varphi \sim g \circ \psi$. Einen Kegel zusammen mit dem Winkel nennt man *Quadrat*:

$$\begin{array}{ccc} X & \xrightarrow[\psi]{} & B \\ \downarrow{\scriptstyle \varphi} & & \downarrow{\scriptstyle g} \\ A & \xrightarrow[f]{} & C \end{array}$$

Wir schreiben $\text{Cone(f, g, X)} :\equiv \sum_{\varphi : X \to A} \sum_{\psi : X \to B} f \circ \varphi \sim g \circ \psi$ für den Typ der X-Kegel über dem gegebenen Winkel.

(b) Der Kegel mit Spitze

$$\text{PB}(f, g) :\equiv \sum_{x:A} \sum_{y:B} f(x) = g(y)$$

und Projektionen $\pi_1 : \text{PB}(f, g) \to A$, $\pi_2 :\equiv \pi_1 \circ \pi_2 : \text{PB}(f, g) \to B$ und der Homotopie $\pi_2 \circ \pi_2 : f \circ \pi_1 \sim g \circ \pi_2$ heißt *Pullback* von f entlang g.

(c) Für einen X-Kegel (φ, ψ, H) heißt die Abbildung

$$V(X, \varphi, \psi, H) :\equiv (x : X) \mapsto (\varphi(x), \psi(x), H_x) : X \to \text{PB}(f, g)$$

Vergleichsabbildung und der X-Kegel zusammen mit dem Winkel heißt *Pullbackquadrat*, wenn $V(X, \varphi, \psi, H)$ eine Äquivalenz ist.

Bemerkung 3.4.5
Sei

$$\begin{array}{ccc} & & B \\ & & \downarrow{\scriptstyle g} \\ A & \xrightarrow[f]{} & C \end{array}$$

ein Winkel.

(a) Der Pullback von f und g vervollständigt den Winkel zu einem Pullbackquadrat.

(b) Sei A punktiert, dann ist der Schleifenraum wie folgt ein Pullback:

$$\begin{array}{ccc} \Omega(A, *) & \longrightarrow & \mathbf{1} \\ \downarrow & & \downarrow{\scriptstyle *} \\ \mathbf{1} & \xrightarrow[*]{} & A \end{array}$$

(c) Wenn f punktiert ist, dann ist die Faser von f wie folgt ein Pullback:

$$\begin{array}{ccc} f^{-1}(*) & \longrightarrow & A \\ \downarrow & & \downarrow{\scriptstyle f} \\ \mathbf{1} & \xrightarrow[*]{} & B \end{array}$$

(d) Die Begriffe Quadrat und Pullbackquadrat sind invariant unter Ersetzen von äquivalenten Typen und homotopen Abbildungen.

Lemma 3.4.6
Sei

$$\begin{array}{ccc} X & \xrightarrow[\psi]{} & B \\ \downarrow{\scriptstyle \varphi} & & \downarrow{\scriptstyle g} \\ A & \xrightarrow[f]{} & C \end{array}$$

ein Quadrat mit Homotopie H, dann sind äquivalent:

(a) Das Quadrat ist ein Pullbackquadrat, bzw. die Vergleichsabbildung $V(X, \varphi, \psi, H) : X \to \mathrm{PB}(f, g)$ ist eine Äquivalenz.

(b) Für jedes Y ist die Abbildung, die eine Funktion $\vartheta : Y \to X$ abbildet auf den Y-Kegel

$$\begin{array}{ccc} Y & \xrightarrow{\psi\circ\vartheta} & B \\ {\scriptstyle \varphi\circ\vartheta}\downarrow & & \downarrow{\scriptstyle g} \\ A & \xrightarrow[f]{} & C \end{array}$$

mit Homotopie H_ϑ, eine Äquivalenz.

(c) Jeder Y-Kegel (φ', ψ', H') faktorisiert eindeutig über X. Das heißt, dass der Typ der Abbildungen $\vartheta : Y \to X$ zusammen mit Homotopien $h : \varphi \circ \vartheta \sim \varphi'$, $k : \psi \circ \vartheta \sim \psi'$ sowie $f(h)^{-1} \cdot H_\vartheta \cdot g(k) = H'$ kontrahierbar ist.

(d) Die induzierte faserweise Abbildung $\prod_{x:A} \varphi^{-1}(x) \to g^{-1}(f(x))$ ist eine faserweise Äquivalenz.

Theorem 3.4.7 (Pullback-Pasting Teil 1)
Seien

$$\begin{array}{ccccc} Y & \longrightarrow & X & \longrightarrow & A \\ \downarrow & & \downarrow & & \downarrow \\ D & \longrightarrow & B & \longrightarrow & C \end{array}$$

jeweils Quadrate, es gebe also entsprechende Homotopien.

(a) Durch Komposition der Homotopien gibt es auch ein drittes Quadrat der Gestalt:

$$\begin{array}{ccc} Y & \longrightarrow & A \\ \downarrow & & \downarrow \\ D & \longrightarrow & C \end{array}$$

(b) Wenn die beiden kleinen Quadrate Pullbacks sind, dann ist auch das dritte, zusammengesetzte "Quadrat" ein Pullback.

Beweis Man benutzt die Äquivalenz zu faserweisen Äquivalenzen. Dann ergibt sich die Aussage durch punktweises Anwenden von 2-aus-3 für Äquivalenzen. □

Theorem 3.4.8 (Fasersequenz)
Sei $(f, p_f) : (A, *) \to^* (B, *)$ eine punktierte Abbildung. Wir verzichten hier auf Angabe der Punktierung und verwenden die Schreibweise $-\Omega f :\equiv p \mapsto f(p^{-1})$. Dann gibt es eine Sequenz:

$$\Omega A \xrightarrow{-\Omega f} \Omega B \longrightarrow f^{-1}(*) \xrightarrow{\pi_1} A \xrightarrow{f} B$$

wobei bei aufeinanderfolgenden Abbildungen stets die linke die Faser der rechten ist - bis auf Äquivalenz. Weiter ist die Abbildung $\Omega B \to f^{-1}(*)$ gegeben durch: $(*, p_f \cdot \_)$.

Beweis Betrache:

$$\begin{array}{ccc} \mathrm{PB}(*,\pi) & \longrightarrow & \mathbf{1} \\ \downarrow & & \downarrow{\scriptstyle *} \\ f^{-1}(*) & \xrightarrow{\pi} & A \\ \downarrow & & \downarrow{\scriptstyle f} \\ \mathbf{1} & \xrightarrow[*]{} & B \end{array}$$

Durch Pullback-Pasting wissen wir, $\mathrm{PB}(*,\pi)$ auch Pullback des zusammengesetzen Quadrats ist. Aber auch ΩB, oder etwas genauer $* = f(*)$ ist Pullback des zusammengesetzen Quadrats, also können wir $\mathrm{PB}(*,\pi)$ äquivalent ersetzen durch ΩB. Wir müssen allerdings etwas genauer hinschauen, um zu sehen, was für eine Abbildung $\Omega B \to f^{-1}(*)$ wir bekommen. Zunächst ist der Pullback:

$$\mathrm{PB}(*,\pi_1) \equiv \left(\sum_{y:\sum_{x:A} f(x)=*} \pi_1(y) = * \right) \simeq \sum_{x:A} (f(x) = *) \times (x = *)$$

Letzterer Typ ist nun äquivalent zu $* = f(*)$ durch: $(x,p,q) \mapsto p^{-1}\cdot f(q)$, da $p^{-1}\cdot f(q)$ der Wert der zusammengesetzten Homotopie im großen Quadrat ist. Die Projektion von $* = f(*)$ nach $f^{-1}(*)$ muss also $(r : * = f(*)) \mapsto (*, r^{-1})$ sein. Die Äquivalenz zu $* = *$ bzw. ΩB ist durch die Punktierung $p_f : f(*) = *$ von f gegeben. Die gesuchte Abbildung ist also $(p : \Omega B) \mapsto (*, p\cdot p_f^{-1})$.

Im so bestimmten Diagram nehmen wir wieder einen Pullback:

$$\begin{array}{ccccc} \mathrm{PB}(\ldots,*) & \longrightarrow & \Omega B & \longrightarrow & \mathbf{1} \\ \downarrow & & \downarrow & & \downarrow{\scriptstyle *} \\ \mathbf{1} & \longrightarrow & f^{-1}(*) & \xrightarrow{\pi_1} & A \\ & & \downarrow & & \downarrow{\scriptstyle f} \\ & & \mathbf{1} & \xrightarrow[*]{} & B \end{array}$$

Genauer ist der neue Pullback: $\mathrm{PB}((*, p_f\cdot\_), *) \equiv \sum_{p:\Omega B} (*, p_f\cdot p) = (*, p_f)$. Durch waagrechtes Pullback-Pasting muss dieser Pullback allerdings äquivalent zu ΩA sein. □

Regeln 3.4.9 (Einhängung)
Zu jedem Typ $A : \mathcal{U}$ gibt es einen Typen $\Sigma A : \mathcal{U}$, die *Suspension* oder *Einhängung* von A. Die Einhängung ist ein höherer induktiver Typ mit folgenden Konstruktoren:

$$\begin{aligned} N &: \Sigma A \\ S &: \Sigma A \\ m &: A \to N =_{\Sigma A} S \end{aligned}$$

ΣA ist durch N punktiert.

Bemerkung 3.4.10
Es gilt: $(\Sigma \mathbf{2}) \simeq S^1$

Beweis Wird noch nachgetragen. □

Definition 3.4.11
Die (induktive) *n-Sphäre* ist rekursiv gegeben durch

$$\begin{aligned} S^0 &:\equiv \mathbf{2} \\ S^{n+1} &:\equiv \Sigma S^n \end{aligned}$$

Da $\mathbf{2}$ punktiert ist und Σ punktierte Typen erhält, können wir S^n als punktierten Typen auffassen.

Regeln 3.4.12
Sei $n \geq -1$. Die n-Abschneidung eines Typen A ist ein höherer induktiver Typ $\|A\|_n : \mathcal{U}$ gegeben durch die folgenden Konstruktoren:

$$\begin{aligned} |\_|_n &: A \to \|A\|_n \\ \mathrm{Nabe} &: (S^{n+1} \to \|A\|_n) \to \|A\|_n \\ \mathrm{Speiche} &: \prod_{s:S^{n+1}\to\|A\|_n} \prod_{x:S^{n+1}} \mathrm{Nabe}(s) = s(x) \end{aligned}$$

Theorem 3.4.13
Sei $A : \mathcal{U}$, dann gilt:

(a) $\|A\|_n$ ist ein n-Typ.

(b) Für n-abgeschnittenes $B : \|A\|_n \to \mathcal{U}$ und $s_0 : \prod_{a:A} B(|a|_n)$ gibt es $s : \prod_{x:\|A\|_n} B(x)$ mit $s(|a|_n) \equiv s_0(a)$.

Wir werden den Beweis in mehreren Schritten führen, die auch für sich genommen interessante Aussagen sind. Zentral für viele Dinge, ist dabei der folgende Satz:

Theorem 3.4.14
Seien X und Y punktierte Typen, dann gibt es eine Äquivalenz:

$$(\Sigma X \to^* Y) \simeq (X \to^* \Omega Y)$$

Beweis Die Äquivalenz lässt sich, unter anderem durch Σ-Rekursion auf die folgende Lücke "??" reduzieren:

$$\begin{aligned}
&\Sigma X \to^* Y \\
\equiv &\sum_{f:\Sigma X \to Y} f(N) = * \\
\simeq &\sum_{\sum_{y_N, y_S : Y} X \to (y_N = y_S)} y_N = * \\
\simeq &\sum_{y_N, y_S : Y} (X \to (y_N = y_S)) \times (y_N = *) \\
\simeq &\sum_{\tilde{y} : \left(\sum_{y_N : Y} y_N = *\right)} \sum_{y_S : Y} (X \to (\pi(\tilde{y}) = y_S)) \\
\simeq &\sum_{y_S : Y} (X \to (* = y_S)) \\
&?? \\
\simeq &\sum_{f : X \to * =_Y *} f(*) = \mathrm{refl}_* \\
\simeq &\sum_{f : X \to \Omega Y} f(*) = \mathrm{refl}_* \\
\simeq &X \to^* \Omega Y
\end{aligned}$$

Nun schließen wir die Lücke. Seien also $y_S : Y$ und $f : X \to * = y_S$. Dann ist

$$\tilde{f} :\equiv (x : X) \mapsto f(x) \cdot f(*)^{-1} : X \to * =_Y *$$

und es gilt $\tilde{f}(*) = \mathrm{refl}_*$. Wir haben also eine Abbildung gefunden. Sei nun $f : X \to * =_Y *$ mit $f(*) = \mathrm{refl}$. Dann ist

$$(*, f) : \sum_{y_S : Y} (X \to (* = y_S))$$

Um zu sehen, dass die Abbildungen invers zueinander sind, müssen wir also noch zeigen:

$$(y_S, f) = (*, \tilde{f})$$

Dazu stellen wir fest, dass der Transport entlang $f(*)$ in $X \to (* = \_)$ punktweise Konkatenation mit $f(*)^{-1}$ ist. Wegen $\mathrm{tr}_{X \to (* = \_)}(f(*))(f) = \tilde{f}$ sind wir also fertig.

Um den Beweis abzuschließen, bleibt also noch zu zeigen, dass für jedes $f : X \to * = *$ und $q : f(*) = \mathrm{refl}_*$ gilt:

$$(f, q) = (\tilde{f}, \dots)$$

wobei "…" eine Gleichheit ist, die durch Gruppoidrechengesetze gegeben ist. Zunächst können wir q benutzen, um eine Homotopie zwischen f und $\tilde{f}$:

$$\mathrm{ap}(f(x) \cdot \_^{-1}, q) : f(x) \cdot f(*)^{-1} = f(x) \cdot \mathrm{refl}_*$$

Bis auf Gruppoidrechengesetze liefert das die benötigte Homotopie. Mit dieser Homotopie haben wir auch eine Gleichheit $s : \tilde{f} = f$. Entlang dieser können wir im abhängigen Typ $g \mapsto g(*) = \mathrm{refl}_*$ transportieren. Als Transport erhalten wir:

$$\mathrm{tr}_{\_(*)=\mathrm{refl}_*}(s) = \mathrm{ap}(\_(*), s)^{-1}\cdot\_$$

Damit die Paare oben gleich sind, muss also gelten: $(\mathrm{ap}(\_(*), s)^{-1}\cdot\_)(\dots) = q$

Zunächst bekommen wir: $\mathrm{ap}(\_(*), s) = \mathrm{ap}(f(*)\cdot\_^{-1}, q)\cdot\dots$, wobei wir wieder "…" für Gruppoidgesetze schreiben. Um Induktion über q anwenden zu können, abstrahieren wir die Situation leicht und verwenden statt $f(*)$ ein beliebiges $p : * = *$. Dafür ergibt sich:

$$\dots\cdot\mathrm{ap}(p\cdot\_^{-1}, q)\cdot\dots = q^{-1}$$

In der Situation oben berechnet sich der Wert des Transports entlang s also wie gewünscht zu q. □

Damit können wir sehen, dass Abbildungen $S^n \to A$ nichts anderes sind als iterierte Schleifen:

Korollar 3.4.15
Für punktiertes A und jedes $n : \mathbb{N}$ gilt:

$$(S^n \to^* A) \simeq \Omega^n A$$

Beweis Wir beweisen per Induktion über n. Für $n \equiv 0$ müssen wir folgende Äquivalenz zeigen:

$$(\mathbf{2} \to^* A) \simeq A$$ □

Da es sich um punktierte Abbildunge handelt, ist die linke Seite nichts anderes als $(\mathbf{1} \to A) \simeq A$. Für den Induktionsschritt verwenden wir Theorem 3.4.14:

$$\begin{aligned}(S^{n+1} \to^* A) &\equiv (\Sigma S^n \to^* A)\\ &\simeq (S^n \to^* \Omega A)\\ &\simeq \Omega^n(\Omega A)\end{aligned}$$

In Lemma 3.4.3 hatten wir gesehen, dass $n+1$-Typen genau die Typen mit n-Typen $\Omega(A, x)$. Das funktioniert in eine Richtung sogar für iterierte Schleifenräume:

Lemma 3.4.16
Sei $n \geq -1$. Ein Typ A ist genau dann ein n-Typ, wenn für alle $x : A$ der Typ $\Omega^{n+1}(A, x)$ kontrahierbar ist.

Beweis Sei $n \equiv -1$. Dann ist zu zeigen: A ist eine Aussage, wenn A kontrahierbar ist. Andererseits kann mit $x : A$ gezeigt werden, dass die Aussage A auch kontrahierbar ist.

Für den Induktionsschritt müssen wir zeigen: A ist genau dann ein $(n+1)$-Typ, wenn $\Omega^{n+2}(A, x)$ für alle $x : A$ kontrahierbar ist. Nach Lemma 3.4.3 reicht es zu zeigen, dass die Kontrahierbarkeit der $\Omega^{n+2}(A, x)$ äquivalent dazu ist, dass jedes $\Omega(A, x)$ ein $n+1$ Typ ist. Nach der Induktionshypothese ist dieser Typ genau dann ein $(n+1)$-Typ, wenn $\Omega^{n+1}(\Omega(A, x), p)$ für jedes $p : \Omega(A, x)$ kontrahierbar ist.

Wir sind also fertig, wenn wir zeigen können: $\Omega^{n+1}(\Omega(A, x), p) \simeq \Omega^{n+2}(A, x)$. Es sollen also die Schleifen $p = p$ äquivalent sein zu den Schleifen $\mathrm{refl}_* = \mathrm{refl}_*$. Nun ist $\_\cdot p^{-1}$ eine Äquivalenz und damit auch:

$$\mathrm{ap}(\_\cdot p^{-1}, \_) : p = p \to p\cdot p^{-1} = p\cdot p^{-1}$$

Letzteres ist durch Einsatz der Gruppoidgesetze äquivalent zu $\mathrm{refl}_* = \mathrm{refl}_*$. □

Korollar 3.4.17
Sei $n \geq -1$. Ein Typ A ist genau dann ein n-Typ, wenn für alle $x : A$, der Typ $(S^{n+1}, *) \to^* (A, x)$ kontrahierbar ist.

Beweis Folgt aus dem Lemma und Korollar 3.4.15. □

Beweis (Theorem 3.4.13) (a) Per Konstruktion ist der Raum der punktierten Abbildungen $S^{n+1} \to^* \|A\|_n$ kontrahierbar: Als Kontraktionszentrum wählen wir die konstante Abbildung $c_* :\equiv x \mapsto *$, die durch refl_* punktiert ist. Für eine punktierte Abbildung $(r, p_r) : (S^{n+1}) \to^* (\|A\|_n, *)$ müssen wir zunächst eine Homotopie $r \sim c_*$ finden. Für $x : S^{n+1}$ sei diese Homotopie gegeben durch

$$\mathrm{Speiche}(x)\cdot\mathrm{Speiche}(*)^{-1}\cdot p_r : r(x) = c_*(x)$$

Für eine Gleichheit von punktierten Abbildungen müssen wir nun noch feststellen:

$$(\mathrm{Speiche}(*)\cdot\mathrm{Speiche}(*)^{-1}\cdot p_r)^{-1}\cdot p_r = \mathrm{refl}_*$$

(b) Zunächst betrachten wir das Induktionsprinzip für $\|A\|_n$. Sei also $B : \|A\|_n \to \mathcal{U}$. Um $s : \prod_{x:\|A\|_n} B(x)$ induktiv zu konstruieren, brauchen wir folgende Daten:

- $s_0 : \prod_{a:A} B(|a|_n)$
- Für alle $r : S^{n+1} \to \|A\|_n$ und $r' : \prod_{x:S^{n+1}} B(r(x))$ ein $N_{r,r'} : B(\text{Nabe(r)})$.
- Für alle $r : S^{n+1} \to \|A\|_n$, $r' : \prod_{x:S^{n+1}} B(r(x))$ und jedes $x : S^{n+1}$ eine abhängige Gleichheit $r(x) =^B_{\text{Speiche}(x)} N_{r,r'}$. □

Die Daten für Naben und Speichen müssen wir also aus der n-Abgeschnittenheit von B herleiten, um das gewünschte Induktionprinzip zu erhalten. Nabe und Speichen in $B(x)$ erhalten wir aber genau aus der n-Abgeschnittenheit der $B(x)$ und korollar 3.4.17.

Bemerkung 3.4.18
Wenn A ein n-Typ ist, dann ist $|\_|_n : A \to \|A\|_n$ eine Äquivalenz.

Beweis Übungsblatt 12. □

Definition 3.4.19
(a) Eine *Gruppe* ist ein 0-Typ G zusammen mit:

- $\_ \cdot \_ : G \to G \to G$
- $\_^{-1} : G \to G$
- $e : G$

und Gleichheiten:

- $\prod_{x,y,z:G} (x \cdot y) \cdot z = x \cdot (y \cdot z)$
- $\prod_{x:G} x \cdot x^{-1} = e = x^{-1} \cdot x$
- $\prod_{x:G} x \cdot e = x = e \cdot x$

(b) Seien G und H Gruppen mit Verknüpfungen $\_ \cdot_G \_$ und $\_ \cdot_H \_$. Eine Abbildung $f : G \to H$ ist ein *Gruppenhomomorphismus*, wenn es

$$\prod_{x,y:G} f(x \cdot_G y) = f(x) \cdot_H f(y)$$

gibt.

Definition 3.4.20
Sei $(A, *)$ punktiert und $n : \mathbb{N}$. Die n-te *Homotopiegruppe* von A ist der Typ

$$\pi_n(A, *) :\equiv \|\Omega^n(A, *)\|_0$$

Für einen weiteren punktierten Typen $(B, *)$ und $(f, p_f) : (A, *) \to^* (B, *)$ ist

$$\pi_n(f, p_f) :\equiv \|\Omega^n(f, p_f)\|_0$$

Beispiel 3.4.21
Wir wissen: $\pi_1(S^1) \simeq \mathbb{Z}$.

Bemerkung 3.4.22
Für einen punktierten Typ $(A, *)$ und $n \geq 1$ ist $\pi_n(A, *)$ stets eine Gruppe. Für einen weiteren punktierten Typen $(B, *)$ und $(f, p_f) : (A, *) \to^* (B, *)$ ist $\pi_n(f, p_f)$ ein Gruppenhomomorphismus.

Definition 3.4.23
Seien A, B punktiert und $f : A \to B$ eine punktierte Abbildung.
(a) Das *Bild* von f ist der folgende Typ:

$$\text{im}(f) :\equiv \sum_{b:B} \|\text{fib}_f(b)\|$$

(b) Der *Kern* von f ist der folgende Typ:

$$\text{Kern}(f) :\equiv \text{fib}_f(*)$$

Theorem 3.4.24
Sei $f : A \to^* B$ eine punktierte Abbildung. Dann gibt es eine lange exakte Sequenz von punktierten 0-Typen:

$$\ldots \longrightarrow \pi_{n+1}(F) \longrightarrow \pi_{n+1}(A) \xrightarrow{(-1)^{n+1}\pi_{n+1}(f)} \pi_{n+1}(B) \longrightarrow \pi_n(F) \longrightarrow \pi_n(A) \xrightarrow{(-1)^n \pi_n(f)} \pi_n(B) \longrightarrow \ldots$$

Beweis Lassen wir hier aus. □

Bemerkung 3.4.25
Man kann auch zeigen, dass es sich für $n \geq 2$ um eine lange exakte Sequenz von abelschen Gruppen handelt. Im Abschnitt mit $n = 1$ ist es eine exakte Sequenz von Gruppen, allerdings im Allgemeinen auch mit Antihomomorphismen.

3.5 Hopf-Faserung

Definition 3.5.1
Ein Typ A heißt *n-zusammenhängend*, wenn $\|A\|_n$ kontrahierbar ist. Im Fall $n = 0$ sagt man auch nur zusammenhängend.

Bemerkung 3.5.2
S^1 ist 0-zusammenhängend und allgemeiner kann man zeigen: S^{n+1} ist n-zusammenhängend.

Beweis Übungsblatt 12: S^1 ist 0-zusammenhängend.
Die Verallgemeinerung auf S^{n+1} lässt sich ähnlich zeigen. □

Wir werden sehen+glauben: Es gibt eine Abbildung $S^3 \to S^2$ mit Faser S^1. Daraus kann man mit der Fasersequenz interessante Schlüsse ziehen.

Definition 3.5.3
Ein *H-Raum* ist ist ein punktierter Typ A mit

- einer Operation $\mu : A \to A \to A$
- Homotopien $\mu(*, \_) \sim \mathrm{id}$ und $\mu(\_, *) \sim \mathrm{id}$

Lemma 3.5.4
Sei A ein zusammenhängender H-Raum. Dann sind die Abbildungen $\mu(x, \_)$ und $\mu(\_, x)$ für alle $x : A$ Äquivalenzen.

Beweis Da wir zeigen wollen, dass etwas eine Äquivalenz für alle $x : A$ ist, wollen wir ein abhängige Funktion in einem -1-abgeschnittenen abhängigen Typen $P : A \to \mathcal{U}$ konstruieren. Äquivalen können wir also auch $P : A \to (-1)$-Type konstruieren. Da (-1)-Type nach bemerkung 3.2.11 ein 0-Typ ist, faktorisiert P über den kontrahierbaren Typ $\|A\|_0$, d.h. wir haben $P' : \|A\|_0 \to (-1)$-Type mit $P' \circ |\_|_0 = P$. Um P' zu zeigen, reicht es das etwa für $|* |_0$ zu machen. Aber $P'(|* |_0)$ ist einfach die Aussage, dass $\mu(*, \_)$ und $\mu(\_, *)$ Äquivalenzen sind und das stimmt, weil sie homotop zur Identität sind. □

Definition 3.5.5
Für einen zusammenhängenden H-Raum A ist die *Hopf-Konstruktion* gegeben als der abhängige Typ $H : \Sigma A \to \mathcal{U}$ mit:

$$H(N) :\equiv A$$
$$H(S) :\equiv A$$
$$H(m(a)) :\equiv \mathrm{ua}(\mu(a, \_))$$

Beispiel 3.5.6
Auf S^1 gibt es eine Gleichheit $p : \mathrm{id} = \mathrm{id}$, gegeben durch eine Homotopie H festgelegt durch $H(*) :\equiv l$ und Nachrechnen des entsprechenden Transports. Auf S^1 können wir eine H-Raum-Struktur rekursiv definieren:

$$\mu(*, \_) :\equiv \mathrm{id}$$
$$\mu(l, \_) :\equiv p$$

Das entspricht der komplexen Multiplikation auf S^1 und definiert eine H-Raum Struktur auf S^1. Der zugehörige abhängige Typ $H_{S^1} : \Sigma S^1 \to \mathcal{U}$ heißt *Hopf-Faserung.*

Fakt 3.5.7
Es gilt $\sum_{x:\Sigma S^1} H_{S^1}(x) \simeq S^3$.

Theorem 3.5.8
Es gilt:
(a) $\Omega^3 S^3 \simeq \Omega^3 S^2$

(b) $\pi_2(S^2) \simeq \mathbb{Z}$

Beweis (Ansatz) Das lässt sich anhand der Fasersequenz für die Hopf-Faserung $S^1 \to S^3 \to S^2$ erkennen. □

3.6 Eilenberg-MacLane Räume

Das Thema dieses Abschnitts wird im HoTT-Buch nur sehr knapp erwähnt und kann im Artikel "Eilenberg-MacLane Spaces in Homotopy Type Theory" von Dan Licata und Eric Finster nachgelesen werden.

Zu einem punktierten Typ A gibt es eine Sequenz von Gruppen, die Homotopiegruppen $\pi_n(A, *)$ für $n \geq 1$. Damit liegt die Frage nahe, ob es auch für jede Sequenz von Gruppen $(G_i)_{i\geq 1}$ einen punktierten Typen A mit $\pi_i(A) \simeq G_i$ gibt. Mit der verständlichen Einschränkung, dass alle G_i für $i \geq 2$ abelsch sind, lässt sich diese Frage positiv beantworten. Wir werden hier zumindest sehen, dass es für jede Gruppe G einen Typ $K(G, 1)$ gibt, für den gilt:

$$\begin{aligned}\pi_0(K(G,1)) &\simeq \mathbf{1}\\ \pi_1(K(G,1)) &\simeq G\\ \pi_n(K(G,1)) &\simeq \mathbf{1} \quad \text{für } n \geq 2\end{aligned}$$

Wir geben direkt einen höheren induktiven Typen an, der das Problem weitgehend löst:

Regeln 3.6.1
Sei G eine Gruppe. Dann ist $\widetilde{K(G,1)}$ der höhere induktive Typ mit Konstruktoren:

$$\begin{aligned}* &: \widetilde{K(G,1)}\\ \mathrm{eq} &: \prod_{g:G} * =_{\widetilde{K(G,1)}} *\\ \mathrm{comp} &: \prod_{g,h:G} \mathrm{eq}(g \cdot h) = \mathrm{eq}(g) \cdot \mathrm{eq}(h)\end{aligned}$$

Definition 3.6.2
Der (erste) *Eilenberg-MacLane Raum* zu einer Gruppe G ist der punktierte Typ

$$K(G,1) :\equiv \|\widetilde{K(G,1)}\|_1$$

Bemerkung 3.6.3
$\| * =_{\widetilde{K(G,1)}} * \|_0 = \Omega K(G,1)$

Beweis Ohne Beweis. □

Fakt 3.6.4
Es gibt eine Äquivalenz

$$\rho : \Omega K(G,1) \simeq G$$

mit $\rho(|\mathrm{eq}(g)|_1) = g$ für alle $g : G$.

3.7 Kohomologie

Die Inhalte in diesem Abschnitt kann man teilweise im Artikel "Cellular Cohomology in Homotopy Type Theory" und in Evan Cavallos Masterarbeit nachlesen. Für jede abelsche Gruppe A gibt es eine Folge von punktierten Typen

$$A, BA, B^2A, B^3A, \ldots$$

sodass $\Omega(B^{n+1}A) = B^nA$ (mit $B^0A \equiv A$) gilt und $B^{n+1}A$ n-zusammenhängend ist.

Definition 3.7.1
Für einen Typ X und eine abelsche Gruppe A ist die n-te *Kohomologiegruppe* von X mit Koeffizienten in A gegeben als:

$$H^n(X, A) :\equiv \|X \to B^n A\|_0$$

Beispiel 3.7.2

(a) Für den Typ $\mathbf{1}$ gilt: $H^0(\mathbf{1}, A) \simeq A$ und $H^{n+1}(\mathbf{1}, A) = \|B^{n+1}A\|_0 \simeq \mathbf{1}$, da $B^{n+1}A$ stets 0-zusammenhängend ist.

(b) Für $\mathbf{2}$ gilt: $H^0(\mathbf{2}, A) \simeq A \times A$ und $H^{n+1}(\mathbf{2}, A) = \|B^{n+1}A\|_0 \times \|B^{n+1}A\|_0 \simeq \mathbf{1}$

(c) Für $A \equiv \mathbb{Z}$ und $X \equiv S^1$ gilt: $H^1(S^1, \mathbb{Z}) = \mathbb{Z}$ (mit etwas Rechnen, siehe Übungsblatt 13).

Bemerkung 3.7.3
Sei $f : X \to Y$. Für festes $n : \mathbb{N}$ ist H^n funktoriell, es gibt also

$$f^* : H^n(Y, A) \to H^n(X, A)$$

und für $g : Y \to Z$ gilt:

$$f^* \circ g^* = (g \circ f)^*$$

Lemma 3.7.4
Für punktierte Typen A und B gilt:

$$(A \to \Omega B) \simeq \Omega(A \to B)$$

wobei $A \to B$ durch die Abbildung $\_ \mapsto *$ punktiert ist.

Beweis

$$\begin{aligned}(A \to \Omega B) &\simeq (A \to * =_B *) \\ &\simeq (\_ \mapsto *) \sim (\_ \mapsto *) \\ &\simeq (\_ \mapsto *) = (\_ \mapsto *) \\ &\simeq \Omega(A \to B)\end{aligned}$$

Bemerkung 3.7.5
Alle Kohomologiegruppen sind abelsch.

Beweis $H^n(X, A) = \|X \to B^n A\|_0 = \|X \to \Omega^2 B^{n+2} A\|_0 = \|\Omega^2(X \to B^{n+2}A)\|_0$ □

Auch hinter der Einhängung steht eine allgemeinere Konstruktion, der sogenannte Pushout:

Regeln 3.7.6 (Pushout)
Seien $A, B, C : \mathcal{U}$ und $f : C \to A$, $g : C \to B$, also folgenden Situation gegeben:

$$\begin{array}{ccc} C & \xrightarrow{f} & A \\ \downarrow{\scriptstyle g} & & \\ B & & \end{array}$$

Dann gibt es einen Typen $A \sqcup_C B :\equiv \mathrm{PO}(f, g) : \mathcal{U}$, den Pushout von f und g, der der höhere induktive Typ mit folgenden Konstruktoren ist:

$$\begin{aligned}\iota_1 &: A \to \mathrm{PO}(f, g) \\ \iota_2 &: B \to \mathrm{PO}(f, g) \\ \mathrm{glue} &: \prod_{c:C} \iota_1(f(c)) = \iota_2(g(c))\end{aligned}$$

Bemerkung 3.7.7
Sei $A : \mathcal{U}$, dann gilt: $\Sigma A \simeq \mathrm{PO}((a : A) \mapsto *, (a : A) \mapsto *)$.

Durch die Rekursion vom Pushout erhält man folgendes:

$$H^1(U \sqcup_{U\cap V} V, A) = \left\| \sum_{f:U\to BA} \sum_{g:V\to BA} \prod_{x:U\cap V} f(x) =_{BA} g(x) \right\|_0$$

Das hilft allerdings noch nicht wirklich beim Berechnen von Kohomologiegruppen. Eine hilfreiche Aussage über Kohomologie von Pushouts ist die Mayer-Vietoris-Sequenz. Auf diese werden wir jetzt zum Abschluss hinarbeiten.

Lemma 3.7.8
Wenn

$$\begin{array}{ccc} X & \longrightarrow & B \\ \downarrow & & \downarrow \\ A & \longrightarrow & C \end{array}$$

ein Pullback ist, dann auch

$$\begin{array}{ccc} X & \xrightarrow{f\circ\pi_1} & C \\ \downarrow & & \downarrow{\scriptstyle \Delta} \\ A\times B & \xrightarrow{f\times g} & C\times C \end{array}$$

Beweis (Idee) Die Kegeltypen sind auf hinreichend gutmütige Art äquivalent:
Die rechte Abbildung im zweiten Pullback lässt sich äquivalent ersetzen durch:

$$(x, y, p) \mapsto p : \sum_{x,y:C} x = y \to C \times C$$

Damit entspricht ein Z-Kegel auf dem zweiten Winkel einer Auswahl eines Paars von Abbildungen $\phi : Z \to A$ und $\psi : Z \to B$ zusammen mit einer abhängigen Abbildung, die für jedes $z : Z$ eine Gleichheit $f(\phi(z)) = g(\psi(z))$ auswählt (bis auf Gleichheit). Das sind dieselben Daten wie für einen Z-Kegel auf dem ersten Winkel. □

Lemma 3.7.9
Sei A punktiert, dann gibt es ein Pullbackquadrat

$$\begin{array}{ccc} \Omega A & \longrightarrow & \mathbf{1} \\ \downarrow{\scriptstyle \Delta} & & \downarrow \\ \Omega A\times\Omega A & \xrightarrow{d} & \Omega A \end{array}$$

Mit $d(p, q) \equiv q\cdot p^{-1}$.

Beweis Es reicht, das für den kanonischen Pullback nachzurechnen:

$$\begin{aligned} &\sum_{(p,q):\Omega A\times\Omega A} \sum_{*:\mathbf{1}} q\cdot p^{-1} = \mathrm{refl}_* \\ \simeq &\sum_{p:\Omega A} \sum_{q:\Omega A} q = p \\ \simeq &\sum_{p:\Omega A} \mathbf{1} \\ \simeq &\Omega A \end{aligned}$$

Lemma 3.7.10
Sei S ein Typ und

$$\begin{array}{ccc} C & \xrightarrow{g} & B \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle \iota_2} \\ A & \xrightarrow[\iota_1]{} & X \end{array}$$

ein Pushout. Dann ist

$$\begin{array}{ccc} (X \to S) & \xrightarrow{\_\circ\iota_2} & (B \to S) \\ {\scriptstyle \_\circ\iota_1}\downarrow & & \downarrow{\scriptstyle \_\circ g} \\ (A \to S) & \xrightarrow[\_\circ f]{} & (C \to S) \end{array}$$

ein Pullback.

Beweis

$$\begin{aligned} & (X \to S) \\ \simeq & \sum_{\phi:A\to S} \sum_{\psi:B\to S} \prod_{c:C} \phi(f(c)) = \psi(g(c)) \\ \simeq & \sum_{\phi:A\to S} \sum_{\psi:B\to S} \phi \circ f = \psi \circ g \\ \simeq & \sum_{\phi:A\to S} \sum_{\psi:B\to S} (\_ \circ f)(\phi) = (\_ \circ g)(\psi) \\ \simeq & \mathrm{PB}(\_ \circ f, \_ \circ g) \end{aligned}$$

Theorem 3.7.11
Für jeden Pushout

$$\begin{array}{ccc} C & \xrightarrow{g} & B \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle \iota_2} \\ A & \xrightarrow[\iota_1]{} & X \end{array}$$

gibt es für jede abelsche Gruppe K eine lange exakte Sequenz von abelschen Gruppen:

$$\begin{array}{ccccc} & & \cdots & \longrightarrow & H^{n-1}(C,K) \\ & \swarrow & & & \\ H^n(X,K) & \longrightarrow & H^n(A,K) \times H^n(B,K) & \xrightarrow{g^*-f^*} & H^n(C,K) \\ & \swarrow & & & \\ H^{n+1}(X,K) & \longrightarrow & H^{n+1}(A,K) \times H^{n+1}(B,K) & \longrightarrow & \cdots \end{array}$$

Beweis Zunächst verwandeln wir den gegebenen Pushout für jedes $n : \mathbb{N}$ in einen Pullback:

$$\begin{array}{ccc} (X \to B^n K) & \xrightarrow{\_\circ\iota_2} & (B \to B^n K) \\ {\scriptstyle \_\circ\iota_1}\downarrow & & \downarrow{\scriptstyle \_\circ g} \\ (A \to B^n K) & \xrightarrow[\_\circ f]{} & (C \to B^n K) \end{array}$$

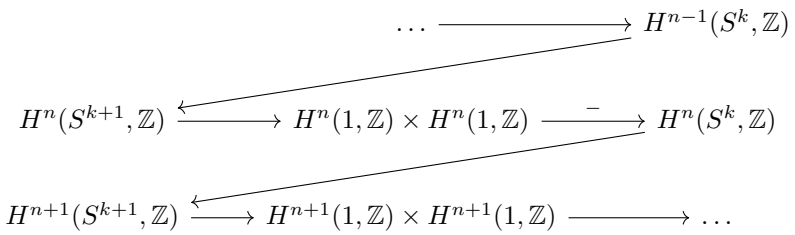
Nun die beiden Lemmata davor anwenden, um per Pullback-Pasting ein großes Faserquadrat zu erhalten:

$$\begin{array}{ccccc} (X \to B^n K) & \longrightarrow & (C \to B^n K) & \longrightarrow & \mathbf{1} \\ {\scriptstyle (\_\circ\iota_1, \_\circ\iota_2)}\downarrow & & \downarrow & & \downarrow \\ (A \to B^n K) \times (B \to B^n K) & \xrightarrow{(\_\circ f)\times(\_\circ g)} & (C \to B^n K) \times (C \to B^n K) & \xrightarrow{d} & (C \to B^n K) \end{array}$$

Darauf können wir die lange exakte Fasersequenz anwenden, um das gewünschte Resultat zu erhalten.□

Korollar 3.7.12
Für $n > 0$ und $k > 0$ gilt $H^n(S^k, \mathbb{Z}) \simeq \mathbb{Z}$ genau dann, wenn $k = n$ und sonst gilt $H^n(S^k, \mathbb{Z}) \simeq \mathbf{1}$.

Beweis (Idee) Einsetzen liefert eine lange exakte Sequenz:



Nun gilt für $n > 0$ stets $H^n(1,\mathbb{Z}) \times H^n(1,\mathbb{Z}) \simeq \mathbf{1} \times \mathbf{1} \simeq \mathbf{1}$, also sind die diagonalen Abbildungen $H^n(S^k,\mathbb{Z}) \to H^{n+1}(S^{k+1},\mathbb{Z})$ jeweils Äquivalenzen. Damit folgt die Aussage aus:

(a) $H^1(S^1,\mathbb{Z}) \simeq \mathbb{Z}$

(b) $H^n(S^1,\mathbb{Z}) \simeq \mathbf{1}$

(c) $H^1(S^k,\mathbb{Z}) \simeq \mathbf{1}$, $k > 1$ □

(a) ist Beispiel 3.7.2 (c).

Für $k \equiv 0$ ergibt sich $H^n(S^0,\mathbb{Z}) \to H^{n+1}(S^1,\mathbb{Z})$. Für $n > 0$ ist aber $H^n(S^0,\mathbb{Z}) \equiv H^n(\mathbf{2},\mathbb{Z}) \simeq \mathbf{1}$ nach Beispiel 3.7.2 (b), was (b) zeigt.

Schließlich folgt (c) aus Bemerkung 3.5.2, also daraus, dass S^k 1-zusammenhängend und S^1 1-abgeschnitten ist:

$$\begin{aligned} H^1(S^k,\mathbb{Z}) &\simeq \|S^k \to S^1\|_0 \\ &\simeq \|\|S^k\|_1 \to S^1\|_0 \\ &\simeq \|\mathbf{1} \to S^1\|_0 \\ &\simeq \|S^1\|_0 \\ &\simeq \mathbf{1} \end{aligned}$$

4 Anhang: Kubische Typentheorien

Es gibt viele unterschiedliche Typentheorien. Wir beschäftigen uns hier hauptsächlich mit der Variante CCHM, zu der man im Artikel "Cubical Type Theory: A constructive interpretation of the univalence axiom" von Cohen, Coquand, Huber und Mörtberg mehr nachlesen kann. Wichtig ist auch die Weiterentwicklung CHM. Diese Typentheorie ist ein Vorfahre der kubischen Elementen der Programmiersprache Agda. In Agda stehen die kubischen Elemente zur Verfügung, wenn es mit "–cubical" ausgeführt wird. Unter "cubical Agda" versteht man nicht etwa eine separate Software, sondern diesen, durch "–cubical" aktivierten, Modus von Agda. Wie dieser verwendet werden kann, kann man sich in der Library "cubical" anschauen, die ausschließlich für diesen Modus von Agda geschrieben wurde.

Eine Warnung vorweg: Die Notation in CCHM und allgemeiner in kubischen Typentheorien weicht teils stark von HoTT ab und ist teilweise gerade für Mathematiker sehr gewöhnungsbedürfitg. Im Folgenden wird zwar größtenteils die Notation der Vorlesung soweit wie möglich verwendet, aber gelegentlich die übliche kubische Notation erwähnt. Zur Übersicht hilft vielleicht auch in Zukunft die folgende Tabelle:

Kubische Typentheorie	Book-HoTT oder Erklärung
Path A u t	kubischer Gleichheitstyp
$u \equiv t$	
cong f p	$\mathrm{ap}(f,p)$
sym p	p^{-1}
transp	Etwas anderes als Transport
subst	Ähnliche Operation wie Transport
$a : A[\varphi \mapsto u]$	Die Einschränkung von a auf den Rand φ ist urteilsgleich u

Tabelle 2: Notation

Kubische Typentheorien (cubical type theories) wurden entwickelt, um die typentheoretischen Mängel der Homotopietypentheorie zu beheben. Zentral ist dabei die Idee, Gleichheiten durch Abbildungen zu modellieren. Allerdings ist das Interval, die Quelle dieser Abbildungen typischerweise kein Typ und daher sind Typen von Gleichheiten auch nicht die Funktionstypen, die wir bisher gesehen haben.

Wohlwissend, dass wir das nicht in der üblichen Art und Weise verwenden dürfen, schreiben wir für Intervallvariablen nun etwa "$i : \mathbf{I}$". Es gibt die folgenden Konstanten und Operationen für Intervallvariablen:

$$0 : \mathbf{I}$$
$$1 : \mathbf{I}$$
$$1 - \varphi : \mathbf{I}$$
$$\varphi \vee \psi : \mathbf{I}$$
$$\varphi \wedge \psi : \mathbf{I}$$

Für diese gelten naheliegende Urteilsgleichheiten, wenn man $\vee$ als Maximum und $\wedge$ als Minimum interpretiert.

Es ist in Ordnung, Intervallvariablen im Kontext zu haben. Der Gleichheitstyp in CCHM wird mit "Path" bezeichnet und hat ähnliche Regeln wie ein Funktionstyps mit der Einschränkung, dass die Quelle stets das Intervall ist und die Bilder der Endpunkte im Typ mit festgehalten sind:

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash u : A \qquad \Gamma \vdash t : A}{\Gamma \vdash \text{Path } A\ u\ t} \qquad \frac{\Gamma \vdash A \qquad \Gamma, i : \mathbf{I} \vdash p(i) : A}{\Gamma \vdash i \mapsto p(i) : \text{Path } A\ p(0)\ p(1)}$$

$$\frac{\Gamma \vdash p : \text{Path } A\ u\ t \qquad \Gamma \vdash r : \mathbf{I}}{\Gamma \vdash p(r) : A} \qquad \frac{\Gamma \vdash A \qquad \Gamma, i : \mathbf{I} \vdash p(i) : A \qquad \Gamma \vdash r : \mathbf{I}}{\Gamma \vdash (i \mapsto p(i))(r) \equiv p(r) : A}$$

$$\frac{\Gamma, i : \mathbf{I} \vdash t(i) \equiv u(i) : A}{\Gamma \vdash t \equiv u : \text{Path } A\ u(0)\ u(1)} \qquad \frac{\Gamma \vdash p : \text{Path } A\ t\ u}{\Gamma \vdash p(0) \equiv t : A} \qquad \frac{\Gamma \vdash p : \text{Path } A\ t\ u}{\Gamma \vdash p(1) \equiv u : A}$$

Im Gegensatz zur induktiven Gleichheit, gibt es hier auch eine Variante namens "PathP", deren "A" mit der Intervallvariable variiert. Das entspricht den abhängigen Gleichheiten aus Abschnitt 3.1.

Damit können wir bereits ein paar einfache Konstruktionen durchführen. Wegen der üblichen Notation und Sprechweise, wollen wir die Elemente von Path-Typen ausnamsweise Pfade nennen.

Definition 4.0.1

(a) Für $a : A$ ist
$$\text{refl}_a :\equiv \text{idp}\ a :\equiv i \mapsto a : \text{Path } A\ a\ a$$

(b) Für einen Pfad $p : \text{Path } A\ t\ u$ gibt es einen *inversen Pfad*:
$$p^{-1} :\equiv \text{sym}\ p :\equiv i \mapsto p(1 - i) : \text{Path } A\ u\ t$$

(c) Wenn $f : A \to B$ eine Funktion ist und $p : \text{Path } A\ t\ u$ ein Pfad, dann ist
$$f(p) :\equiv \text{cong}\ f\ p :\equiv i \mapsto f(p(i))$$

Die Konkatenation fehlt hier, weil sie soweit noch nicht definiert werden kann. Dazu brauchen wir eine sogenannte Komposition. Terme und Typen die von n Intervallvariablen abhängen, stellt man sich als auf einem n-dimensionalen Würfel definiert vor. Komposition und andere Operationen erlauben es, aus Termen und Typen, die auf speziellen Teilen des Rands eines n-Würfels gegeben sind, Terme und Typen auf dem ganzen Würfel zu konstruieren.

Um über Ränder von n-Würfeln zu reden, verwenden wir *Randformeln* oder einfach *Ränder*. Für eine Intervallvariable $i : \mathbf{I}$ gibt es die Ränder

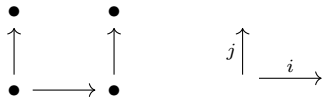
$$(i = 0) \quad \text{und} \quad (i = 1)$$

Für Ränder gibt es wie für Intervallvariablen die Operationen $\vee$ und $\wedge$, die hier Vereinigung und Schnitt bedeuten. Weiter gibt es Konstanten $0_{\mathbf{F}}$ und $1_{\mathbf{F}}$. Das "$\mathbf{F}$" steht dabei für Face-Lattice. Es gelten naheliegende Gleichungen, wie zum Beispiel $(i = 0) \wedge (i = 1) = 0_{\mathbf{F}}$.

Damit können Unterpolyeder des Randes eines n-Würfels beschrieben werden, zum Beispiel

$$(i = 0) \vee (i = 1) \vee (j = 0)$$

Für den Rand eines Quadrats ohne "Deckel", also eines "offenen Quadrats":



Randformeln darf man in den Kontext aufnehmen, was dann bedeutet, dass die Formeln "gelten". Für zwei Pfade $p : \text{Path } A\ t\ u$ und $q : \text{Path } A\ u\ v$ ist es möglich, die folgenden Urteile zu fällen:

$$i : \mathbf{I}, j : \mathbf{I}, (j = 0) \vdash p(i) : A$$
$$i : \mathbf{I}, j : \mathbf{I}, (i = 1) \vdash q(j) : A$$
$$i : \mathbf{I}, j : \mathbf{I}, (i = 0) \vdash t : A$$

Die Ränder aus diesen Urteilen können geschnitten werden, es gibt also etwa den Rand $(i = 1) \wedge (j = 0)$ der in den beiden Rändern, $(i = 1)$ und $(j = 0)$ enthalten ist. Wir müssen nun Prüfen, dass die Urteile oben einen wohldefinierten Term auf dem gesamten Rand ergeben. Tatsächlich ist das der Fall:

$$i : \mathbf{I}, j : \mathbf{I}, (j = 0) \wedge (i = 1) \vdash p(1) \equiv q(0) : A$$
$$i : \mathbf{I}, j : \mathbf{I}, (j = 0) \wedge (i = 0) \vdash p(0) \equiv t : A$$

Das bedeutet, dass wir einen Term auf dem offenen Quadrat von oben gefunden haben:

$$\begin{array}{ccc} t & & v \\ {\scriptstyle \text{refl}_t}\uparrow & & \uparrow{\scriptstyle q(j)} \\ t & \xrightarrow[p(i)]{} & u \end{array}$$

In CCHM bzw Agda gibt es in dieser Situation einen Term

$$i \mapsto \text{hcomp}^j\ [(i = 0) \mapsto t, (i = 1) \mapsto q(j)]\ (p(i)) : \text{Path } A\ t\ v$$

Die hier vorgestellten Regeln sind bei weitem noch nicht alles, was eine kubische Typentheorie ausmacht. Um das Univalenzaxiom bzw Univalenztheorem beweisen zu können, reicht das bisherige nicht aus.

Index