

Extensionale Interpretation des Kategorischen Imperativs

Cornelius Diekmann

November 27, 2022

Abstract

Language warning: German ahead.

Extensionale Interpretation des Kategorischen Imperativs. Persönliche Interpretation basierend auf Sekundärliteratur.

Beispiel referenz: [1]

Contents

1	Schnelleinstieg Isabelle/HOL	3
1.1	Typen	3
1.2	Beweise	3
1.3	Mehr Typen	3
1.4	Funktionen	3
1.5	Mengen	4
2	Disclaimer	4
2.1	Über den Titel	5
3	Handlung	5
3.1	Interpretation: Gesinnungsethik vs. Verantwortungsethik	7
4	Kant's Kategorischer Imperativ	8
5	Beispiel Person	8
6	Maxime	9
6.1	Maxime in Sinne Kants?	9
6.2	Die Goldene Regel	10
6.3	Maximen Debugging	11
6.4	Beispiel	12
6.5	Maximen Kombinieren	13
7	Schleier des Nichtwissens	14
7.1	Wohlgeformte Handlungsabsicht	15
7.2	Wohlgeformte Maxime	18

8	Kategorischer Imperativ	18
8.1	Triviale Maximen die den Kategorischen Imperativ immer Erfüllen	21
8.2	Zusammenhang Goldene Regel	21
8.3	Maximen die den Kategorischen Imperativ immer Erfüllen	22
9	Ausführbarer Beispielgenerator	23
9.1	Kombination vom Maximen	24
9.1.1	Konjunktion	24
9.1.2	Disjunktion	25
10	Utilitarismus	27
10.1	Goldene Regel und Utilitarismus im Einklang	28
11	Zahlenwelt Helper	29
12	Beispiel: Zahlenwelt	31
12.1	Ungültige Handlung	32
12.2	Nicht-Wohlgeformte Handlungen	32
12.3	Wohlgeformte Handlungen	33
12.4	Maxime für individuellen Fortschritt	34
12.4.1	Einzellbeispiele	35
12.5	Maxime für allgemeinen Fortschritt	36
12.6	Maxime für strikten individuellen Fortschritt	37
12.7	Maxime für globales striktes Optimum	37
12.8	Maxime für globales Optimum	38
12.9	Ungültige Maxime	39
13	Beispiel: Zahlenwelt2	39
14	Gesetz	47
15	Experimental: Moralisch Gesetzes Ableiten	48
15.1	Allgemeines Gesetz Ableiten	48
15.2	Implementierung Moralisch ein Allgemeines Gesetz Ableiten	49
16	Gesetze	50
16.1	Case Law Absolut	50
16.2	Case Law Relativ	50
17	Simulation	51
18	Beispiel: BeispielZahlenwelt aber mit Gesetz (Experimental)	52
18.1	Setup	52
18.2	Beispiele	52
19	Einkommensteuergesetzgebung	55

20 Beispiel: Steuern	58
20.1 Setup für Beispiele	60
20.2 Beispiel: Keiner Zahlt Steuern	60
20.3 Beispiel: Ich zahle 1 Steuer	60
20.4 Beispiel: Jeder zahle 1 Steuer	60
20.5 Beispiel: Vereinfachtes Deutsches Steuersystem	60
21 Vereinfachtes Deutsches Steuersystem vs. die Steuermaxime	61

1 Schnelleinstieg Isabelle/HOL

1.1 Typen

Typen werden per $::$ annotiert. Beispielsweise sagt $3::nat$, dass 3 eine natürliche Zahl (nat) ist.

1.2 Beweise

Die besondere Fähigkeit im Beweisassistent Isabelle/HOL liegt darin, maschinengeprüfte Beweise zu machen.

Beispiel:

lemma $\langle 3 = 2+1 \rangle$

In der PDFversion wird der eigentliche Beweis ausgelassen. Aber keine Sorge, der Computer hat den Beweis überprüft. Würde der Beweis nicht gelten, würde das PDF garnicht compilieren.

Ich wurde schon für meine furchtbaren Beweise zitiert. Ist also ganz gut, dass wir nur Ergebnisse im PDF sehen und der eigentliche Beweis ausgelassen ist. Am besten kann man Beweise sowieso im Isabelle Editor anschauen und nicht im PDF.

1.3 Mehr Typen

Jeder Typ der mit einem einfachen Anführungszeichen anfängt ist ein polymorpher Typ. Beispiel: $'a$ oder $'\alpha$. So ein Typ ist praktisch ein generischer Typ, welcher durch jeden anderen Typen instanziiert werden kann.

Beispielsweise steht $'nat$ für einen beliebigen Typen, während nat der konkrete Typ der natürlichen Zahlen ist.

Wenn wir nun $3::'a$ schreiben handelt es sich nur um das generische Numeral 3. Das ist so generisch, dass z.B. noch nicht einmal die Plusoperation darauf definiert ist. Im Gegensatz dazu ist $3::nat$ die natürliche Zahl 3, mit allen wohlbekannten Rechenoperationen. Im Beweis obigen **lemma** $\langle 3 = 2+1 \rangle$ hat Isabelle die Typen automatisch inferiert.

1.4 Funktionen

Beispiel: Eine Funktionen welche eine natürliche Zahl nimmt und eine natürliche Zahl zurück gibt ($nat \Rightarrow nat$):

fun *beispielfunktion* :: $\langle nat \Rightarrow nat \rangle$ **where**
 $\langle beispielfunktion\ n = n + 10 \rangle$

Funktionsaufrufe funktionieren ohne Klammern.

lemma $\langle beispielfunktion\ 32 = 42 \rangle$

Funktionen sind gecurried. Hier ist eine Funktion welche 2 natürliche Zahlen nimmt und eine natürliche Zahl zurück gibt ($nat \Rightarrow nat \Rightarrow nat$):

fun *addieren* :: $\langle nat \Rightarrow nat \Rightarrow nat \rangle$ **where**
 $\langle addieren\ a\ b = a + b \rangle$

lemma $\langle addieren\ 32\ 10 = 42 \rangle$

Currying bedeutet auch, wenn wir *addieren* nur mit einem Argument aufrufen (welches eine natürliche Zahl *nat* sein muss), dass wir eine Funktion zurückbekommen, die noch das zweite Argument erwartet, bevor sie das Ergebnis zurückgeben kann.

Beispiel: *addieren 10::nat $\Rightarrow nat$*

Zufälligerweise ist *addieren 10* equivalent zu *beispielfunktion*:

lemma $\langle addieren\ 10 = beispielfunktion \rangle$

Zusätzlich lassen sich Funktionen im Lambda Calculus darstellen. Beispiel:

lemma $\langle (\lambda n::nat. n+10)\ 3 = 13 \rangle$

lemma $\langle beispielfunktion = (\lambda n. n+10) \rangle$

1.5 Mengen

Mengen funktionieren wie normale mathematische Mengen.

Beispiel. Die Menge der geraden Zahlen:

lemma $\langle \{0,2,4,6,8,10,12\} \subseteq \{n::int. n \bmod 2 = 0\} \rangle$

2 Disclaimer

Ich habe

- wenig Ahnung von Philosophie.
- keine Ahnung von Recht und Jura.
- und schon gar keine Ahnung von Strafrecht oder Steuerrecht.

Und in dieser Session werden ich all das zusammenwerfen. Dies ist ein instabiler Development Snapshot. Er enthält sinnvolles und weniger sinnvolle Experimente!

Cheers!

2.1 Über den Titel

Der Titel lautet *Extensionale Interpretation des Kategorischen Imperativs*. Dabei sind die Wörter wie folgt zu verstehen

- *Extensional* bezieht sich hier auf den Fachbegriff der Logik <https://en.wikipedia.org/wiki/Extensionality>, welcher besagt, dass Objekte gleich sind, wenn sie die gleichen externen Eigenschaften aufweisen. Beispielsweise sind zwei Funktionen gleich, wenn sie für alle Eingaben die gleiche Ausgabe liefern: $(f = g) = (\forall x. f\ x = g\ x)$. Die interne (intensionale) Implementierung der Funktionen mag unterschiedlich sein, dennoch sind sie gleich. Dies ist die natürliche Gleichheit in HOL, welche uns erlaubt unser Modell bequem zu shallow-embedden. Meine extensionale Modellierung prägt diese Theorie stark. Beispielsweise sind Handlungen extensional modelliert, d.h nur die äußerlich messbaren Ergebnisse werden betrachtet. Dies widerspricht vermutlich stark Kants Vorstellung.
- *Interpretation* besagt, dass es sich hier um meine persönliche Interpretation handelt. Diese Theorie ist keine strenge Formalisierung der Literatur, sondern enthält sehr viele persönliche Meinungen.
- *Kategorischer Imperativ* bezieht sich auf Kants Kategorischer Imperativ. Ziel dieser Theorie ist es, moralische Entscheidungen basierend auf Kants Idee zu machen.

3 Handlung

Beschreibt Handlungen als Änderung der Welt. Unabhängig von der handelnden Person. Wir beschreiben nur vergangene bzw. mögliche Handlungen und deren Auswirkung.

Eine Handlung ist reduziert auf deren Auswirkung. Intention oder Wollen ist nicht modelliert, da wir irgendwie die geistige Welt mit der physischen Welt verbinden müssen und wir daher nur messbare Tatsachen betrachten können.

Handlungen können Leute betreffen. Handlungen können aus Sicht Anderer wahrgenommen werden. Ich brauche nur Welt vorher und Welt nachher. So kann ich handelnde Person und beobachtende Person trennen.

datatype *'world handlung* = *Handlung* (*vorher*: *<'world>*) (*nachher*: *<'world>*)

definition *ist-noop* :: *<'world handlung* \Rightarrow *bool* **where**
<ist-noop h \equiv *vorher h* = *nachher h*

Handlung als Funktion gewrapped. Diese abstrakte Art eine Handlung zu modelliert so ein bisschen die Absicht oder Intention.

datatype (*'person*, *'world*) *handlungsabsicht* = *Handlungsabsicht* \langle *'person* \Rightarrow *'world* \Rightarrow *'world option* \rangle

Eine (*'person*, *'world*) *handlungsabsicht* gibt eine *'world option* zurück, anstatt einer *'world*. Handlungsabsichten sind damit partielle Funktionen, was modelliert, dass die Ausführung einer Handlungsabsicht scheitern kann. Beispielsweise könnte ein Dieb versuchen ein Opfer zu bestehlen; wenn sich allerdings kein passendes Opfer findet, dann darf die Handlung scheitern. Oder es könnte der pathologische Sonderfall eintreten, dass ein Dieb sich selbst bestehlen soll. Auch hier darf die Handlung scheitern. Von außen betrachtet ist eine solche gescheiterte Handlung nicht zu unterscheiden vom Nichtstun. Allerdings ist es für die moralische Betrachtung dennoch wichtig zu unterscheiden, ob die Handlungsabsicht ein gescheiterter Diebstahl war, oder ob die Handlungsabsicht einfach Nichtstun war. Dadurch dass Handlungsabsichten partiell sind, können wir unterscheiden ob die Handlung wie geplant ausgeführt wurde oder gescheitert ist. Moralisch sind Stehlen und Nichtstun sehr verschieden.

fun *nachher-handeln* :: \langle *'person* \Rightarrow *'world* \Rightarrow (*'person*, *'world*) *handlungsabsicht* \Rightarrow *'world* \rangle
where
 \langle *nachher-handeln handelnde-person welt* (*Handlungsabsicht h*) =
 (case *h handelnde-person welt of Some welt' \Rightarrow welt'*
 | *None \Rightarrow welt*) \rangle

Die Funktion *nachher-handeln* besagt, dass eine gescheiterte Handlung die Welt nicht verändert. Ab diesem Punkt sind also die Handlungen "sich selbst bestehlen" und "Nichtstun" von außen ununterscheidbar, da beide die Welt nicht verändern.

definition *handeln* :: \langle *'person* \Rightarrow *'world* \Rightarrow (*'person*, *'world*) *handlungsabsicht* \Rightarrow *'world handlung* \rangle
where
 \langle *handeln handelnde-person welt ha* \equiv *Handlung welt* (*nachher-handeln handelnde-person welt ha*) \rangle

Die Funktion *nachher-handeln* liefert die Welt nach der Handlung. Die Funktion *handeln* liefert eine *'world handlung*, welche die Welt vor und nach der Handlung darstellt.

Beispiel, für eine Welt die nur aus einer Zahl besteht: Wenn die Zahl kleiner als 9000 ist erhöhe ich sie, ansonsten schl'gt die Handlung fehl.

definition \langle *beispiel-handlungsabsicht* \equiv *Handlungsabsicht* (λ - *n*. if *n* < 9000 then *Some* (*n*+1) else *None*) \rangle

lemma *nachher-handeln "Peter" (42::nat) beispiel-handlungsabsicht* = 43
lemma *handeln "Peter" (42::nat) beispiel-handlungsabsicht* = *Handlung* 42 43
lemma *nachher-handeln "Peter" (9000::nat) beispiel-handlungsabsicht* = 9000
lemma *ist-noop* (*handeln "Peter" (9000::nat) beispiel-handlungsabsicht*)

Von Außen können wir Funktionen nur extensional betrachten, d.h. Eingabe und Ausgabe anschauen. Die Absicht die sich in einer Funktion verstecken kann ist schwer zu erkennen. Dies deckt sich ganz gut damit, dass Isabelle standardmäßig Funktionen nicht printet. Eine (*'person*, *'world*) *handlungsabsicht* kann nicht geprinted werden!

Da Funktionen nicht geprinted werden können, sieht *beispiel-handlungsabsicht* so aus: *Handlungsabsicht* -

Um eine gescheiterte Handlung von einer Handlung welche die Welt nicht verändert zu unterscheiden, sagen wir, dass eine handlungsabsicht ausführbar ist, wenn die ausgeführte Handlungsabsicht nicht gescheitert ist:

```
fun ausfuehrbar :: 'person  $\Rightarrow$  'world  $\Rightarrow$  ('person, 'world) handlungsabsicht  $\Rightarrow$  bool
where
  ausfuehrbar p welt (Handlungsabsicht h) = (h p welt  $\neq$  None)
```

Nicht ausführbare Handlungen resultieren in unserem Modell im Nichtstun:

```
lemma nicht-ausfuehrbar-ist-noop:
   $\langle \neg \text{ausfuehrbar } p \text{ welt } ha \implies \text{ist-noop } (\text{handeln } p \text{ welt } ha) \rangle$ 
```

3.1 Interpretation: Gesinnungsethik vs. Verantwortungsethik

Sei eine Ethik eine Funktion, welche einem beliebigen α eine Bewertung Gut = *True*, Schlecht = *False* zuordnet.

- Eine Ethik hat demnach den Typ: $\alpha \Rightarrow \text{bool}$.

Laut <https://de.wikipedia.org/wiki/Gesinnungsethik> ist eine Gesinnungsethik "[...] eine der moralischen Theorien, die Handlungen nach der Handlungsabsicht [...] bewertet, und zwar ungeachtet der nach erfolgter Handlung eingetretenen Handlungsfolgen."

- Demnach ist eine Gesinnungsethik: $(\text{'person}, \text{'world}) \text{ handlungsabsicht} \Rightarrow \text{bool}$.

Nach <https://de.wikipedia.org/wiki/Verantwortungsethik> steht die Verantwortungsethik dazu im strikten Gegensatz, da die Verantwortungsethik "in der Bewertung des Handelns die Verantwortbarkeit der *tatsächlichen Ergebnisse* betont."

- Demnach ist eine Verantwortungsethik: $\text{'world handlung} \Rightarrow \text{bool}$.

Da *handeln* eine Handlungsabsicht $(\text{'person}, \text{'world}) \text{ handlungsabsicht}$ in eine konkrete Änderung der Welt 'world handlung überführt, können wie die beiden Ethiktypen miteinander in Verbindung setzen. Wir sagen, eine Gesinnungsethik und eine Verantwortungsethik sind konsistent, genau dann wenn für jede Handlungsabsicht, die Gesinnungsethik die Handlungsabsicht genau so bewertet, wie die Verantwortungsethik die Handlungsabsicht bewerten würde, wenn die die Handlungsabsicht in jeder möglichen Welt und als jede mögliche handelnde Person tatsächlich ausführt wird und die Folgen betrachtet werden:

```
definition gesinnungsethik-verantwortungsethik-konsistent
  ::  $\langle ((\text{'person}, \text{'world}) \text{ handlungsabsicht} \Rightarrow \text{bool}) \Rightarrow (\text{'world handlung} \Rightarrow \text{bool}) \Rightarrow \text{bool} \rangle$  where
   $\langle \text{gesinnungsethik-verantwortungsethik-konsistent } \text{gesinnungsethik } \text{verantwortungsethik} \equiv$ 
     $\forall \text{ handlungsabsicht.}$ 
```

gesinnungsethik handlungsabsicht \longleftrightarrow
 $(\forall \text{ person welt. verantwortungsethik } (\text{handeln person welt handlungsabsicht})) \rangle$

Ich habe kein Beispiel für eine Gesinnungsethik und eine Verantwortungsethik, die tatsächlich konsistent sind.

4 Kant's Kategorischer Imperativ



Immanuel Kant

„Handle nur nach derjenigen *Maxime*, durch die du zugleich wollen kannst, dass sie ein allgemeines Gesetz werde.“

https://de.wikipedia.org/wiki/Kategorischer_Imperativ

Meine persönliche, etwas utilitaristische, Interpretation.

5 Beispiel Person

Eine Beispielbevölkerung.

datatype *person* = *Alice* | *Bob* | *Carol* | *Eve*

Unsere Bevölkerung ist sehr endlich:

lemma *UNIV-person*: $\langle \text{UNIV} = \{ \text{Alice}, \text{Bob}, \text{Carol}, \text{Eve} \} \rangle$

Wir werden unterscheiden:

- *'person*: generischer Typ, erlaubt es jedes Modell einer Person und Bevölkerung zu haben.
- *person*: Unser minimaler Beispieltyp, bestehend aus *Alice*, *Bob*, ...

6 Maxime

Nach <https://de.wikipedia.org/wiki/Maxime> ist eine Maxime ein persönlicher Grundsatz des Wollens und Handelns. Nach Kant ist eine Maxime ein "subjektives Prinzip des Wollens".

Modell einer *Maxime*: Eine Maxime in diesem Modell beschreibt ob eine Handlung in einer gegebenen Welt gut ist.

Faktisch ist eine Maxime

- *'person*: die handelnde Person, i.e., *ich*.
- *'world handlung*: die zu betrachtende Handlung.
- *bool*: Das Ergebnis der Betrachtung. *True* = Gut; *False* = Schlecht.

Wir brauchen sowohl die *'world handlung* als auch die *'person* aus deren Sicht die Maxime definiert ist, da es einen großen Unterschied machen kann ob ich selber handel, ob ich Betroffener einer fremden Handlung bin, oder nur Außenstehender.

datatype *('person, 'world) maxime = Maxime* $\langle 'person \Rightarrow 'world\ handlung \Rightarrow bool \rangle$

Auswertung einer Maxime:

fun *okay* :: $\langle ('person, 'world)\ maxime \Rightarrow 'person \Rightarrow 'world\ handlung \Rightarrow bool \rangle$ **where**
 $\langle okay\ (Maxime\ m)\ p\ h = m\ p\ h \rangle$

Beispiel

definition *maxime-mir-ist-alles-recht* :: $\langle ('person, 'world)\ maxime \rangle$ **where**
 $\langle maxime-mir-ist-alles-recht \equiv Maxime\ (\lambda -.\ True) \rangle$

6.1 Maxime in Sinne Kants?

Kants kategorischer Imperativ ist eine deontologische Ethik, d.h., "Es wird eben nicht bewertet, was die Handlung bewirkt, sondern wie die Absicht beschaffen ist." https://de.wikipedia.org/wiki/Kategorischer_Imperativ.

Wenn wir Kants kategorischen Imperativ bauen wollen, dürfen wir also nicht die Folgen einer Handlung betrachten, sondern nur die Absicht dahinter. Doch unsere *Maxime* betrachtet eine *'world handlung*, also eine konkrete Handlung, die nur durch ihre Folgen gegeben ist. Die Maxime betrachtet keine Handlungsabsicht (*'person, 'world handlungsabsicht*).

Kant unterscheidet unter Anderem "zwischen »apriorischen« und »empirischen« Urteilen" [1]. Wenn wir uns den Typ *'world handlung* als Beobachtung der Welt *vorher* und *nachher* anschauen, dann könnte man sagen, unser Moralbegriff der *Maxime* sei empirisch. Für Kant gilt jedoch: "Alle Moralbegriffe [...] haben *a priori* ihren Sitz und Ursprung ausschließlich in der Vernunft" [1]. Hier widerspricht unser Modell wieder Kant, da unser Modell empirisch ist und nicht apriorisch.

Dies mag nun als Fehler in unserem Modell verstanden werden. Doch irgendwo müssen wir praktisch werden. Nur von Handlungsabsichten zu reden, ohne dass die beabsichtigten Folgen betrachtet werden ist mir einfach zu abstrakt und nicht greifbar.

Alles ist jedoch nicht verloren, denn "Alle rein mathematischen Sätze sind [...] apriorisch" [1]. Und auch Russel schlussfolgert: "Um ein ausreichendes Kriterium zu gewinnen, müßten wir Kants rein formalen Standpunkt aufgeben und die Wirkung der Handlungen in Betracht ziehen" [1].

Auch Kants kategorischer Imperativ und die Goldene Regel sind grundverschieden: <https://web.archive.org/web/20220123174117/https://www.goethegymnasium-hildesheim.de/index.php/faecher/faecher/gesellschaftswissenschaften/philosophie> Dennoch, betrachten wir den kategorischen Imperativ als eine Verallgemeinerung der goldenen Regel.

6.2 Die Goldene Regel

Die Goldene Regel nach https://de.wikipedia.org/wiki/Goldene_Regel sagt:

„Behandle andere so, wie du von ihnen behandelt werden willst.“

„Was du nicht willst, dass man dir tu, das füg auch keinem andern zu.“

So wie wir behandelt werden wollen ist modelliert durch eine $(\text{'person}, \text{'world}) \text{ maxime}$.

Die goldene Regel testet ob eine Handlung, bzw. Handlungsabsicht moralisch ist. Um eine Handlung gegen eine Maxime zu testen fragen wir uns:

- Was wenn jeder so handeln würde?
- Was wenn jeder nach dieser Maxime handeln würde?

Beispielsweise mag "stehlen" und "bestohlen werden" die gleiche Handlung sein, jedoch wird sie von Täter und Opfer grundverschieden wahrgenommen.

definition *bevoelkerung* :: $\langle \text{'person set} \rangle$ **where** $\langle \text{bevoelkerung} \equiv UNIV \rangle$

definition *wenn-jeder-so-handelt*

:: $\langle \text{'world} \Rightarrow (\text{'person}, \text{'world}) \text{ handlungsabsicht} \Rightarrow (\text{'world handlung}) \text{ set} \rangle$

where

$\langle \text{wenn-jeder-so-handelt welt handlungsabsicht} \equiv$

$(\lambda \text{handelnde-person. handeln handelnde-person welt handlungsabsicht}) \text{' bevoelkerung} \rangle$

fun *was-wenn-jeder-so-handelt-aus-sicht-von*

:: $\langle \text{'world} \Rightarrow (\text{'person}, \text{'world}) \text{ maxime} \Rightarrow (\text{'person}, \text{'world}) \text{ handlungsabsicht} \Rightarrow \text{'person} \Rightarrow \text{bool} \rangle$

where

$\langle \text{was-wenn-jeder-so-handelt-aus-sicht-von welt m handlungsabsicht betroffene-person} =$

$(\forall h \in \text{wenn-jeder-so-handelt welt handlungsabsicht. okay m betroffene-person h}) \rangle$

Für eine gegebene Welt und eine gegebene Maxime nennen wir eine Handlungsabsicht genau dann moralisch, wenn die Handlung auch die eigene Maxime erfüllt, wenn die Handlung von anderen durchgeführt würde.

definition *moralisch* ::

$\langle \text{'world} \Rightarrow (\text{'person}, \text{'world}) \text{ maxime} \Rightarrow (\text{'person}, \text{'world}) \text{ handlungsabsicht} \Rightarrow \text{bool} \rangle$ **where**

$\langle \text{moralisch welt handlungsabsicht maxime} \equiv$

$\forall p \in \text{bevoelkerung. was-wenn-jeder-so-handelt-aus-sicht-von welt handlungsabsicht maxime p} \rangle$

Faktisch bedeutet diese Definition, wir bilden das Kreuzprodukt Bevölkerung x Bevölkerung, wobei jeder einmal als handelnde Person auftritt und einmal als betroffene Person.

lemma *moralisch-unfold*:

$\langle \text{moralisch welt } (Maxime\ m) \text{ handlungsabsicht} \longleftrightarrow$
 $(\forall p1 \in \text{bevoelkerung}. \forall p2 \in \text{bevoelkerung}. m\ p1\ (\text{handeln } p2\ \text{welt handlungsabsicht})) \rangle$

lemma $\langle \text{moralisch welt } (Maxime\ m) \text{ handlungsabsicht} \longleftrightarrow$

$(\forall (p1, p2) \in \text{bevoelkerung} \times \text{bevoelkerung}. m\ p1\ (\text{handeln } p2\ \text{welt handlungsabsicht})) \rangle$

lemma *moralisch-simp*:

$\langle \text{moralisch welt } m \text{ handlungsabsicht} \longleftrightarrow$
 $(\forall p1\ p2. \text{okay } m\ p1\ (\text{handeln } p2\ \text{welt handlungsabsicht})) \rangle$

Wir können die goldene Regel auch umformulieren, nicht als Imperativ, sondern als Beobachtung eines Wunschzustandes: Wenn eine Handlung für eine Person okay ist, dann muss sie auch Okay sein, wenn jemand anderes diese Handlung ausführt.

Formal: $m\ \text{ich}\ (\text{handeln } \text{ich}\ \text{welt handlungsabsicht}) \implies \forall p2. m\ \text{ich}\ (\text{handeln } p2\ \text{welt handlungsabsicht})$

Genau dies können wir aus unserer Definition von *moralisch* ableiten:

lemma *goldene-regel*:

$\langle \text{moralisch welt } m \text{ handlungsabsicht} \implies$
 $\text{okay } m\ \text{ich}\ (\text{handeln } \text{ich}\ \text{welt handlungsabsicht}) \implies$
 $\forall p2. \text{okay } m\ \text{ich}\ (\text{handeln } p2\ \text{welt handlungsabsicht}) \rangle$

Für das obige lemma brauchen wir die Annahme $m\ \text{ich}\ (\text{handeln } \text{ich}\ \text{welt handlungsabsicht})$ gar nicht. Wenn für eine gegebene *Maxime* m eine Handlungsabsicht moralisch ist, dann ist es auch okay, wenn ich von der Handlungsabsicht betroffen bin, egal wer sie ausführt.

corollary

$\langle \text{moralisch welt } m \text{ handlungsabsicht} \implies$
 $\forall p2. \text{okay } m\ \text{ich}\ (\text{handeln } p2\ \text{welt handlungsabsicht}) \rangle$

Die umgekehrte Richtung gilt nicht, weil diese Formulierung nur die Handlungen betrachtet, die okay sind.

Hier schlägt das Programmiererherz höher: Wenn *'person* aufzählbar ist haben wir ausführbaren Code: *moralisch* = *moralisch-exhaust enum-class.enum* wobei *moralisch-exhaust* implementiert ist als *moralisch-exhaust bevoelk welt maxime handlungsabsicht* \equiv *case maxime of Maxime m \Rightarrow list-all* $(\lambda(p, x). m\ p\ (\text{handeln } x\ \text{welt handlungsabsicht}))$ (*List.product bevoelk bevoelk*).

6.3 Maximen Debugging

Der folgende Datentyp modelliert ein Beispiel in welcher Konstellation eine gegebene Maxime verletzt ist:

datatype *'person opfer* = *Opfer* $\langle 'person \rangle$
datatype *'person taeter* = *Taeter* $\langle 'person \rangle$
datatype $('person, 'world) \text{ verletzte-maxime} =$
VerletzteMaxime

$\langle 'person\ Opfer \rangle$ — verletzt für; das Opfer
 $\langle 'person\ taeter \rangle$ — handelnde Person; der Täter
 $\langle 'world\ handlung \rangle$ — Die verletzende Handlung

Die folgende Funktion liefert alle Gegebenheiten welche eine Maxime verletzen:

```

fun debug-maxime
  ::  $\langle ('world \Rightarrow 'printable-world) \Rightarrow 'world \Rightarrow$ 
     $( 'person, 'world) maxime \Rightarrow ( 'person, 'world) handlungsabsicht$ 
     $\Rightarrow (( 'person, 'printable-world) verletzte-maxime) set \rangle$ 
where
   $\langle debug-maxime\ print-world\ welt\ m\ handlungsabsicht =$ 
    { VerletzteMaxime
      (Opfer p1) (Taeter p2)
      (map-handlung print-world (handeln p2 welt handlungsabsicht)) | p1 p2.
       $\neg okay\ m\ p1\ (handeln\ p2\ welt\ handlungsabsicht) \} \rangle$ 

```

Es gibt genau dann keine Beispiele für Verletzungen, wenn die Maxime erfüllt ist:

lemma $\langle debug-maxime\ print-world\ welt\ maxime\ handlungsabsicht = \{ \}$
 $\longleftrightarrow moralisch\ welt\ maxime\ handlungsabsicht \rangle$

6.4 Beispiel

Beispiel: Die Welt sei nur eine Zahl und die zu betrachtende Handlungsabsicht sei, dass wir diese Zahl erhöhen. Die Mir-ist-alles-Recht Maxime ist hier erfüllt:

lemma $\langle moralisch$
 $(42::nat)$
 $maxime-mir-ist-alles-recht$
 $(Handlungsabsicht\ (\lambda(person::person)\ welt.\ Some\ (welt + 1))) \rangle$

Beispiel: Die Welt ist modelliert als eine Abbildung von Person auf Besitz. Die Maxime sagt, dass Leute immer mehr oder gleich viel wollen, aber nie etwas verlieren wollen. In einer Welt in der keiner etwas hat, erfüllt die Handlung jemanden 3 zu geben die Maxime.

lemma $\langle moralisch$
 $[Alice \mapsto (0::nat), Bob \mapsto 0, Carol \mapsto 0, Eve \mapsto 0]$
 $(Maxime\ (\lambda person\ handlung.$
 $(the\ ((vorher\ handlung)\ person)) \leq (the\ ((nachher\ handlung)\ person))))$
 $(Handlungsabsicht\ (\lambda person\ welt.\ Some\ (welt(person \mapsto 3)))) \rangle$

lemma $\langle debug-maxime\ show-map$
 $[Alice \mapsto (0::nat), Bob \mapsto 0, Carol \mapsto 0, Eve \mapsto 0]$
 $(Maxime\ (\lambda person\ handlung.$
 $(the\ ((vorher\ handlung)\ person)) \leq (the\ ((nachher\ handlung)\ person))))$
 $(Handlungsabsicht\ (\lambda person\ welt.\ Some\ (welt(person \mapsto 3))))$
 $= \{ \} \rangle$

Wenn nun *Bob* allerdings bereits 4 hat, würde die obige Handlung ein Verlust für ihn bedeuten und die Maxime ist nicht erfüllt.

lemma $\langle \neg \text{moralisch}$
 $[Alice \mapsto (0::nat), Bob \mapsto 4, Carol \mapsto 0, Eve \mapsto 0]$
 $(\text{Maxime } (\lambda person \text{ handlung.}$
 $(\text{the } ((\text{vorher handlung } person)) \leq (\text{the } ((\text{nachher handlung } person))))$
 $(\text{Handlungsabsicht } (\lambda person \text{ welt. Some } (\text{welt}(person \mapsto 3))))) \rangle$

lemma $\langle \text{debug-maxime show-map}$
 $[Alice \mapsto (0::nat), Bob \mapsto 4, Carol \mapsto 0, Eve \mapsto 0]$
 $(\text{Maxime } (\lambda person \text{ handlung.}$
 $(\text{the } ((\text{vorher handlung } person)) \leq (\text{the } ((\text{nachher handlung } person))))$
 $(\text{Handlungsabsicht } (\lambda person \text{ welt. Some } (\text{welt}(person \mapsto 3)))))$
 $= \{ \text{VerletzteMaxime } (\text{Opfer Bob}) (\text{Taeter Bob})$
 $(\text{Handlung } [(Alice, 0), (Bob, 4), (Carol, 0), (Eve, 0)])$
 $[(Alice, 0), (Bob, 3), (Carol, 0), (Eve, 0)]) \}$

6.5 Maximen Kombinieren

Konjunktion (Und) zweier Maximen.

fun *MaximeConj*
 $:: ('person, 'welt) \text{ maxime} \Rightarrow ('person, 'welt) \text{ maxime} \Rightarrow ('person, 'welt) \text{ maxime}$
where
 $\text{MaximeConj } (\text{Maxime } m1) (\text{Maxime } m2) = \text{Maxime } (\lambda p \ h. m1 \ p \ h \wedge m2 \ p \ h)$

lemma *okay-MaximeConj*: $\text{okay } (\text{MaximeConj } m1 \ m2) \ p \ h \longleftrightarrow \text{okay } m1 \ p \ h \wedge \text{okay } m2 \ p \ h$

lemma *moralisch-MaximeConj*:
 $\text{moralisch welt } (\text{MaximeConj } m1 \ m2) \ ha \longleftrightarrow \text{moralisch welt } m1 \ ha \wedge \text{moralisch welt } m2 \ ha$

lemma *moralisch-MaximeConj-False*:
 $\text{moralisch welt } (\text{MaximeConj } m1 \ (\text{Maxime } (\lambda - . \text{True}))) \ ha \longleftrightarrow \text{moralisch welt } m1 \ ha$

lemma *moralisch-MaximeConj-True*:
 $\neg \text{moralisch welt } (\text{MaximeConj } m1 \ (\text{Maxime } (\lambda - . \text{False}))) \ ha$

Disjunktion (Oder) zweier Maximen.

fun *MaximeDisj*
 $:: ('person, 'welt) \text{ maxime} \Rightarrow ('person, 'welt) \text{ maxime} \Rightarrow ('person, 'welt) \text{ maxime}$
where
 $\text{MaximeDisj } (\text{Maxime } m1) (\text{Maxime } m2) = \text{Maxime } (\lambda p \ h. m1 \ p \ h \vee m2 \ p \ h)$

lemma *okay-MaximeDisj*: $\text{okay } (\text{MaximeDisj } m1 \ m2) \ p \ h \longleftrightarrow \text{okay } m1 \ p \ h \vee \text{okay } m2 \ p \ h$

Leider ist *MaximeDisj* weniger schön, weil es kein genau-dann-wenn mit der Disjunktion ($m1 \vee m2$) gibt.

lemma *moralisch-MaximeDisjI*:
 $\text{moralisch welt } m1 \ ha \vee \text{moralisch welt } m2 \ ha \implies \text{moralisch welt } (\text{MaximeDisj } m1 \ m2) \ ha$

Rückrichtung gilt leider nicht.

lemma *moralisch-MaximeDisj1*:

moralisch welt m1 ha \implies moralisch welt (MaximeDisj m1 m2) ha

lemma *moralisch-MaximeDisj2*:

moralisch welt m2 ha \implies moralisch welt (MaximeDisj m1 m2) ha

lemma *moralisch-MaximeDisj-False*:

moralisch welt (MaximeDisj m1 (Maxime (λ - -. False))) ha \longleftrightarrow moralisch welt m1 ha

lemma *moralisch-MaximeDisj-True*:

moralisch welt (MaximeDisj m1 (Maxime (λ - -. True))) ha

7 Schleier des Nichtwissens

Rawls' Schleier des Nichtwissens https://de.wikipedia.org/wiki/Schleier_des_Nichtwissens ist ein fiktives Modell, » über die zukünftige Gesellschaftsordnung entscheiden können, aber selbst nicht wissen, an welcher Stelle dieser Ordnung sie sich später befinden werden, also unter einem „Schleier des Nichtwissens“ stehen.« Quote wikipedia

Wir bedienen uns bei der Idee dieses Modells um gültige Handlungsabsichten und Maximen zu definieren. Handlungsabsichten und Maximen sind nur gültig, wenn darin keine Personen hardge-coded werden.

Beispielsweise ist folgende Handlungsabsicht ungültig: *λ ich welt. if ich = Alice then Do-A welt else Do-B welt*

Handlungsabsichten und Maximen müssen immer generisch geschrieben werden, so dass die handelnden und betroffenen Personen niemals anhand ihres Namens ausgewählt werden.

unser Modell von Handlungsabsichten und Maximen stellt beispielsweise die handelnde Person als Parameter bereit. Folgendes ist also eine gültige Handlung: *λ ich welt. ModifiziereWelt welt ich*

Auch ist es erlaubt, Personen in einer Handlungsabsicht oder Maxime nur anhand ihrer Eigenschaften in der Welt auszuwählen. Folgendes wäre eine wohlgeformte Handlung, wenn auch eine moralisch fragwürdige: *λ ich welt. enteignen ‘ {opfer. besitz ich < besitz opfer} ’*

Um diese Idee von wohlgeformten Handlungsabsichten und Maximen zu formalisieren bedienen wir uns der Idee des Schleiers des Nichtwissens. Wir sagen, dass Handlungsabsichten wohlgeformt sind, wenn die Handlungsabsicht gleich bleibt, wenn man sowohl die handelnde Person austauscht, als auch alle weltlichen Eigenschaften dieser Person. Anders ausgedrückt: Wohlgeformte Handlungsabsichten und Maximen sind solche, bei denen bei der Definition noch nicht feststeht, auf we sie später zutreffen.

Für jede Welt muss eine Welt-Personen Swap (wps) Funktion bereit gestellt werden, die alle Weltlichen Eigenschaften von 2 Personen vertauscht:

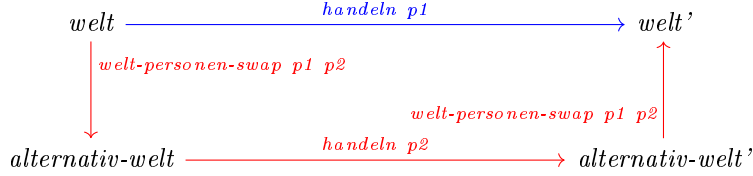
type-synonym (*'person*, *'world*) *wp-swap* = $\langle 'person \Rightarrow 'person \Rightarrow 'world \Rightarrow 'world \rangle$

Ein jeder (*'person*, *'world*) *wp-swap* sollte mindestens folgendes erfüllen:

definition $wps-id :: ('person, 'world) wp-swap \Rightarrow 'world \Rightarrow bool$

where

$wps-id\ wps\ welt \equiv \forall p1\ p2. wps\ p2\ p1\ (wps\ p1\ p2\ welt) = welt$



7.1 Wohlgeformte Handlungsabsicht

fun *wohlgeformte-handlungsabsicht*

$:: \langle ('person, 'world) wp-swap \Rightarrow 'world \Rightarrow ('person, 'world) handlungsabsicht \Rightarrow bool \rangle$

where

$\langle wohlgeformte-handlungsabsicht\ wps\ welt\ (Handlungsabsicht\ h) =$
 $(\forall p1\ p2. h\ p1\ welt = map-option\ (wps\ p2\ p1)\ (h\ p2\ (wps\ p1\ p2\ welt))) \rangle$

declare *wohlgeformte-handlungsabsicht.simps*[simp del]

lemma *wohlgeformte-handlungsabsicht-ausfuehrbar*:

$wohlgeformte-handlungsabsicht\ wps\ welt\ ha \implies$
 $\forall p1\ p2. ausfuehrbar\ p1\ welt\ ha \longleftrightarrow ausfuehrbar\ p2\ (wps\ p1\ p2\ welt)\ ha$

lemma *wohlgeformte-handlungsabsicht-mit-wpsid*:

$\langle wohlgeformte-handlungsabsicht\ wps\ welt\ ha \implies$
 $wps-id\ wps\ welt \implies$
 $\forall p1\ p2. handeln\ p1\ welt\ ha =$
 $map-handlung\ (wps\ p2\ p1)\ (handeln\ p2\ (wps\ p1\ p2\ welt)\ ha) \rangle$

Folgende Folgerung erklärt die Definition vermutlich besser:

lemma *wohlgeformte-handlungsabsicht-wpsid-imp-handeln*:

$\langle wohlgeformte-handlungsabsicht\ wps\ welt\ ha \implies wps-id\ wps\ welt \implies$
 $(\forall p1\ p2. handeln\ p1\ welt\ ha =$
 $Handlung\ welt$
 $(wps\ p2\ p1\ (nachher-handeln\ p2\ (wps\ p1\ p2\ welt)\ ha))) \rangle$

lemma *wfh-handeln-imp-wpsid*:

$(\forall p1\ p2. handeln\ p1\ welt\ ha =$
 $map-handlung\ (wps\ p2\ p1)\ (handeln\ p2\ (wps\ p1\ p2\ welt)\ ha)) \implies$
 $wps-id\ wps\ welt$

lemma *wohlgeformte-handlungsabsicht-wpsid-simp*:

$wohlgeformte-handlungsabsicht\ wps\ welt\ ha \wedge wps-id\ wps\ welt$
 \longleftrightarrow
 $(\forall p1\ p2. ausfuehrbar\ p1\ welt\ ha \longleftrightarrow ausfuehrbar\ p2\ (wps\ p1\ p2\ welt)\ ha)$
 $\wedge (\forall p1\ p2. handeln\ p1\ welt\ ha =$

$map\text{-}handlung\ (wps\ p2\ p1)\ (handeln\ p2\ (wps\ p1\ p2\ welt)\ ha))$

fun *wohlgeformte-handlungsabsicht-gegenbeispiel*
 $:: \langle ('person, 'world)\ wp\text{-}swap \Rightarrow 'world \Rightarrow ('person, 'world)\ handlungsabsicht \Rightarrow 'person \Rightarrow 'person \Rightarrow bool \rangle$
where
 $\langle wohlgeformte\text{-}handlungsabsicht\text{-}gegenbeispiel\ wps\ welt\ (Handlungsabsicht\ h)\ taeter\ opfer \longleftrightarrow$
 $h\ taeter\ welt \neq map\text{-}option\ (wps\ opfer\ taeter)\ (h\ opfer\ (wps\ taeter\ opfer\ welt)) \rangle$

lemma $\langle wohlgeformte\text{-}handlungsabsicht\text{-}gegenbeispiel\ wps\ welt\ ha\ p1\ p2 \implies$
 $\neg wohlgeformte\text{-}handlungsabsicht\ wps\ welt\ ha \rangle$

Nach der gleichen Argumentation müssen Maxime und Handlungsabsicht so generisch sein, dass sie in allen Welten zum gleichen Ergebnis kommen.

definition *maxime-und-handlungsabsicht-generalisieren*
 $:: \langle ('person, 'world)\ wp\text{-}swap \Rightarrow 'world \Rightarrow$
 $('person, 'world)\ maxime \Rightarrow ('person, 'world)\ handlungsabsicht \Rightarrow 'person \Rightarrow bool \rangle$
where
 $\langle maxime\text{-}und\text{-}handlungsabsicht\text{-}generalisieren\ wps\ welt\ m\ h\ p =$
 $(\forall p1\ p2. (ausfuehrbar\ p\ welt\ h \wedge ausfuehrbar\ p\ (wps\ p1\ p2\ welt)\ h)$
 $\longrightarrow okay\ m\ p\ (handeln\ p\ welt\ h) \longleftrightarrow okay\ m\ p\ (handeln\ p\ (wps\ p1\ p2\ welt)\ h)) \rangle$

Für eine gegebene Maxime schließt die Forderung *maxime-und-handlungsabsicht-generalisieren* leider einige Handlungen aus. Beispiel: In einer Welt besitzt *Alice* 2 und *Eve* hat 1 Schulden. Die Maxime ist, dass Individuen gerne keinen Besitz verlieren. Die Handlung sei ein globaler reset, bei dem jeden ein Besitz von 0 zugeordnet wird. Leider generalisiert diese Handlung nicht, da *Eve* die Handlung gut findet, *Alice* allerdings nicht.

lemma
 $\langle \neg maxime\text{-}und\text{-}handlungsabsicht\text{-}generalisieren$
 $swap$
 $((\lambda x. 0)(Alice := (2::int), Eve := -1))$
 $(Maxime\ (\lambda ich\ h. (vorher\ h)\ ich \leq (nachher\ h)\ ich))$
 $(Handlungsabsicht\ (\lambda ich\ w. Some\ (\lambda -. 0)))$
 $Eve \rangle$

Die Maxime und $('person, 'world)\ wp\text{-}swap$ müssen einige Eigenschaften erfüllen. Wir kürzen das ab mit *wpsm*: Welt Person Swap Maxime.

Die Person für die Maxime ausgewertet wird und swappen der Personen in der Welt muss equivalent sein:

definition *wpsm-kommutiert*
 $:: \langle ('person, 'world)\ maxime \Rightarrow ('person, 'world)\ wp\text{-}swap \Rightarrow 'world \Rightarrow bool \rangle$
where
 $\langle wpsm\text{-}kommutiert\ m\ wps\ welt \equiv$
 $\forall\ p1\ p2\ ha.$

$okay\ m\ p2\ (handeln\ p1\ (wps\ p1\ p2\ welt)\ ha)$
 \longleftrightarrow
 $okay\ m\ p1\ (Handlung\ welt\ (wps\ p1\ p2\ (nachher-handeln\ p1\ (wps\ p2\ p1\ welt)\ ha))))\rangle$

lemma *wpsm-kommutiert-handlung-raw*:

$\langle wpsm-kommutiert\ m\ wps\ welt =$
 $(\forall\ p1\ p2\ ha.$
 $okay\ m\ p2\ (Handlung\ (wps\ p1\ p2\ welt)\ (nachher-handeln\ p1\ (wps\ p1\ p2\ welt)\ ha))$
 \longleftrightarrow
 $okay\ m\ p1\ (Handlung\ welt\ (wps\ p1\ p2\ (nachher-handeln\ p1\ (wps\ p2\ p1\ welt)\ ha))))\rangle$

lemma *wpsm-kommutiert-unfold-handlungsabsicht*:

$\langle wpsm-kommutiert\ m\ wps\ welt =$
 $(\forall\ p1\ p2\ ha.$
 $okay\ m\ p2\ (handeln\ p1\ (wps\ p1\ p2\ welt)\ ha)$
 \longleftrightarrow
 $okay\ m\ p1\ (handeln\ p1\ welt\ (Handlungsabsicht\ (\lambda p\ w.\ Some\ (wps\ p1\ p2\ (nachher-handeln\ p\ (wps\ p2\ p1\ w)\ ha))))))$
 \rangle

Wenn sowohl *wohlgeformte-handlungsabsicht* als auch *wpsm-kommutiert*, dann erhalten wir ein sehr intuitives Ergebnis, welches besagt, dass ich handelnde Person und Person für die die Maxime gelten soll vertauschen kann.

lemma *wfh-wpsm-kommutiert-simp*:

assumes *wpsid*: *wps-id wps welt*
shows $\langle wohlgeformte-handlungsabsicht\ wps\ welt\ ha \implies$
 $wpsm-kommutiert\ m\ wps\ welt \implies$
 $okay\ m\ p2\ (handeln\ p1\ (wps\ p1\ p2\ welt)\ ha)$
 \longleftrightarrow
 $okay\ m\ p1\ (handeln\ p2\ welt\ ha) \rangle$

Die Rückrichtung gilt auch, aber da wir das für alle Handlungsabsichten in der Annahme brauchen, ist das eher weniger hilfreich.

lemma *wfh-kommutiert-wpsm*:

assumes *wpsid*: *wps-id wps welt*
shows
 $\langle \forall\ ha.\ wohlgeformte-handlungsabsicht\ wps\ welt\ ha \wedge$
 $(\forall\ p1\ p2.\ okay\ m\ p2\ (handeln\ p1\ (wps\ p1\ p2\ welt)\ ha)$
 \longleftrightarrow
 $okay\ m\ p1\ (handeln\ p2\ welt\ ha)) \implies$
 $wpsm-kommutiert\ m\ wps\ welt \rangle$

lemma *wpsm-kommutiert-map-handlung*:

assumes *wpsid*: *wps-id wps welt*
and *wps-sym*: *wps p1 p2 welt = wps p2 p1 welt*

shows $\langle wpsm\text{-}kommutiert\ m\ wps\ (wps\ p1\ p2\ welt) \implies$
 $okay\ m\ p1\ (map\text{-}handlung\ (wps\ p1\ p2)\ (handeln\ p1\ welt\ ha))$
 \longleftrightarrow
 $okay\ m\ p2\ (handeln\ p1\ welt\ ha) \rangle$

7.2 Wohlgeformte Maxime

definition *wohlgeformte-maxime-auf*

$:: \langle 'world\ handlung \Rightarrow ('person, 'world)\ wp\text{-}swap \Rightarrow ('person, 'world)\ maxime \Rightarrow bool \rangle$

where

$\langle wohlgeformte\text{-}maxime\text{-}auf\ h\ wps\ m \equiv$
 $\forall p1\ p2. okay\ m\ p1\ h \longleftrightarrow okay\ m\ p2\ (map\text{-}handlung\ (wps\ p1\ p2)\ h) \rangle$

definition *wohlgeformte-maxime*

$:: \langle ('person, 'world)\ wp\text{-}swap \Rightarrow ('person, 'world)\ maxime \Rightarrow bool \rangle$

where

$\langle wohlgeformte\text{-}maxime\ wps\ m \equiv$
 $\forall h. wohlgeformte\text{-}maxime\text{-}auf\ h\ wps\ m \rangle$

Beispiel:

lemma $\langle wohlgeformte\text{-}maxime\ swap\ (Maxime\ (\lambda ich\ h. (vorher\ h)\ ich \leq (nachher\ h)\ ich)) \rangle$

8 Kategorischer Imperativ

Wir haben mit der goldenen Regel bereits definiert, wann für eine gegebene Welt und eine gegebene maxime, eine Handlungsabsicht moralisch ist:

- $moralisch :: 'world \Rightarrow ('person, 'world)\ maxime \Rightarrow ('person, 'world)\ handlungsabsicht \Rightarrow bool$

Effektiv testet die goldene Regel eine Handlungsabsicht.

Nach meinem Verständnis generalisiert Kant mit dem Kategorischen Imperativ diese Regel, indem die Maxime nicht mehr als gegeben angenommen wird, sondern die Maxime selbst getestet wird. Sei die Welt weiterhin gegeben, dass müsste der kategorische Imperativ folgende Typsignatur haben:

- $'world \Rightarrow ('person, 'world)\ maxime \Rightarrow bool$

Eine Implementierung muss dann über alle möglichen Handlungsabsichten allquantifizieren.

Ich behaupte, der kategorischer Imperativ lässt sich wie folgt umformulieren:

- Handle nur nach derjenigen Maxime, durch die du zugleich wollen kannst, dass sie ein allgemeines Gesetz werde.
- Handle nur nach derjenigen Maxime, durch die du zugleich wollen kannst, dass sie jeder befolgt, im Sinne der goldenen Regel.

- Handle nur nach derjenigen Maxime, durch die du zugleich wollen kannst, dass sie (Handlung+Maxime) moralisch ist.
- Wenn es jemanden gibt der nach einer Maxime handeln will, dann muss diese Handlung nach der Maxime moralisch sein.
- Für jede Handlungsabsicht muss gelten: Wenn jemand in jeder Welt nach der Handlungsabsicht handeln würde, dann muss diese Handlung moralisch sein.

Daraus ergibt sich diese Formalisierung:

Für eine bestimmte Handlungsabsicht: Wenn es eine Person gibt für die diese Handlungsabsicht moralisch ist, dann muss diese Handlungsabsicht auch für alle moralisch (im Sinne der goldenen Regel) sein.

definition *kategorischer-imperativ-auf*

$:: \langle ('person, 'world) handlungsabsicht \Rightarrow 'world \Rightarrow ('person, 'world) maxime \Rightarrow bool \rangle$

where

$\langle kategorischer-imperativ-auf\ ha\ welt\ m \equiv$

$(\exists ich. ausfuehrbar\ ich\ welt\ ha \wedge okay\ m\ ich\ (handeln\ ich\ welt\ ha)) \longrightarrow moralisch\ welt\ m\ ha \rangle$

Für alle möglichen (wohlgeformten) Handlungsabsichten muss dies nun gelten:

definition *kategorischer-imperativ*

$:: \langle ('person, 'world) wp-swap \Rightarrow 'world \Rightarrow ('person, 'world) maxime \Rightarrow bool \rangle$

where

$\langle kategorischer-imperativ\ wps\ welt\ m \equiv$

$\forall ha. wohlgeformte-handlungsabsicht\ wps\ welt\ ha \longrightarrow$
 $kategorischer-imperativ-auf\ ha\ welt\ m \rangle$

Wir führen die interne Hilfsdefinition *kategorischer-imperativ-auf* ein um den kategorischen Imperativ nur für eine Teilmenge aller Handlungen besser diskutieren zu können.

TODO: Leider fehlen mir Beispiele von Maximen welche den kategorischen Imperativ uneingeschränkt auf allen Handlungsabsichten erfüllen.

Diese $\neg ist-noop\ (handeln\ ich\ welt\ h)$ gefällt mir gar nicht. Wir brauchen es aber, damit die Beispiele funktionieren. Das ist nötig, um pathologische Grenzfälle auszuschließen. Beispielsweise ist von-sich-selbst stehlen eine no-op. No-ops sind normalerweise nicht böse. Stehlen ist schon böse. Dieser Grenzfall in dem Stehlen zur no-op wird versteckt also den Charakter der Handlungsabsicht und muss daher ausgeschlossen werden. Ganz glücklich bin ich mit der Rechtfertigung aber nicht. Eventuell wäre es schöner, Handlungen partiell zu machen, also dass Handlungsabsichten auch mal *None* zurückgeben dürfen. Das könnte einiges rechtfertigen. Beispielsweise ist Stehlen: jemand anderen etwas wegnehmen. Nicht von sich selbst. Allerdings machen partielle Handlungen alles komplizierter.

In der Definition ist *wohlgeformte-handlungsabsicht* ein technisch notwendiges Implementierungsdetail um nicht-wohlgeformte Handlungen auszuschließen.

Minimal andere Formulierung:

lemma

$\langle \textit{kategorischer-imperativ wps welt m} \longleftrightarrow$
 $(\forall ha.$
 $(\exists p.$
 $\textit{wohlgeformte-handlungsabsicht wps welt ha} \wedge$
 $\textit{ausfuehrbar p welt ha} \wedge$
 $\textit{okay m p (handeln p welt ha)})$
 $\longrightarrow \textit{moralisch welt m ha}) \rangle$

Der Existenzquantor lässt sich auch in einen Allquantor umschreiben:

lemma

$\langle \textit{kategorischer-imperativ wps welt m} \longleftrightarrow$
 $(\forall ha \textit{ ich.}$
 $\textit{wohlgeformte-handlungsabsicht wps welt ha} \wedge \textit{ausfuehrbar ich welt ha} \wedge$
 $\textit{okay m ich (handeln ich welt ha)} \longrightarrow \textit{moralisch welt m ha}) \rangle$

Vergleich zu *moralisch*. Wenn eine Handlung moralisch ist, dann impliziert diese Handlung die Kernforderung des *kategorischer-imperativ*. Wenn die Handlungsabsicht für mich okay ist, ist sie auch für alle anderen okay.

lemma $\langle \textit{moralisch welt m ha} \implies$
 $\textit{kategorischer-imperativ-auf ha welt m} \rangle$

Die andere Richtung gilt nicht, z.B. ist die Maxime die immer False zurückgibt ein Gegenbeispiel.

lemma $\langle m = \textit{Maxime} (\lambda - . \textit{False}) \implies$
 $\textit{kategorischer-imperativ-auf ha welt m} \longrightarrow \textit{moralisch welt m ha}$
 $\implies \textit{False} \rangle$

Für jede Handlungsabsicht: wenn ich so handeln würde muss es auch okay sein, wenn zwei beliebige Personen so handeln, wobei einer Täter und einer Opfer ist.

lemma *kategorischer-imperativ-simp:*

$\langle \textit{kategorischer-imperativ wps welt m} \longleftrightarrow$
 $(\forall ha \textit{ p1 p2 ich.}$
 $\textit{wohlgeformte-handlungsabsicht wps welt ha} \wedge \textit{ausfuehrbar ich welt ha} \wedge$
 $\textit{okay m ich (handeln ich welt ha)})$
 $\longrightarrow \textit{okay m p1 (handeln p2 welt ha)}) \rangle$

Introduction rules

lemma *kategorischer-imperativI:*

$\langle (\bigwedge ha \textit{ ich p1 p2. wohlgeformte-handlungsabsicht wps welt ha} \implies$
 $\textit{ausfuehrbar ich welt ha} \implies$
 $\textit{okay m ich (handeln ich welt ha)} \implies \textit{okay m p1 (handeln p2 welt ha)})$
 $\implies \textit{kategorischer-imperativ wps welt m} \rangle$

lemma *kategorischer-imperativ-auf1*:

$$\begin{aligned} &\langle (\bigwedge \text{ich } p1 \text{ } p2. \text{ ausfuehrbar ich welt ha} \\ &\implies \text{okay m ich (handeln ich welt ha)} \implies \text{okay m } p1 \text{ (handeln } p2 \text{ welt ha)}) \\ &\implies \text{kategorischer-imperativ-auf ha welt m} \rangle \end{aligned}$$

Um den *kategorischer-imperativ-auf* einer Handlungsabsicht zu zeigen muss entweder die Handlungsabsicht moralisch sein, oder es darf keine Person geben, die diese Handlung auch tatsächlich unter gegebener Maxime ausführen würde:

lemma *kategorischer-imperativ-auf2*:

$$\begin{aligned} &\langle \text{moralisch welt m ha} \vee \neg(\exists \text{ } p. \text{ ausfuehrbar } p \text{ welt ha} \wedge \text{okay m } p \text{ (handeln } p \text{ welt ha)}) \\ &\longleftrightarrow \text{kategorischer-imperativ-auf ha welt m} \rangle \end{aligned}$$

8.1 Triviale Maximen die den Kategorischen Imperativ immer Erfüllen

Die Maxime die keine Handlung erlaubt (weil immer False) erfüllt den kategorischen Imperativ:

lemma $\langle \text{kategorischer-imperativ wps welt (Maxime } (\lambda \text{ich h. False})) \rangle$

Allerdings kann mit so einer Maxime nie etwas moralisch sein.

lemma $\langle \neg \text{moralisch welt (Maxime } (\lambda \text{ich h. False})) \text{ } h \rangle$

Die Maxime die jede Handlung erlaubt (weil immer True) erfüllt den kategorischen Imperativ:

lemma $\langle \text{kategorischer-imperativ wps welt (Maxime } (\lambda \text{ich h. True})) \rangle$

Allerdings ist mit so einer Maxime alles moralisch.

lemma $\langle \text{moralisch welt (Maxime } (\lambda \text{ich h. True})) \text{ } h \rangle$

8.2 Zusammenhang Goldene Regel

Mit der goldenen Regel konnten wir wie folgt moralische Entscheidungen treffen: $\llbracket \text{moralisch welt m handlungsabsicht; okay m ich (handeln ich welt handlungsabsicht)} \rrbracket \implies \forall p2. \text{okay m ich (handeln } p2 \text{ welt handlungsabsicht)}$

In Worten: Wenn eine Handlungsabsicht moralisch ist (nach goldener Regel) und es okay ist für mich diese Handlung auszuführen, dann ist es auch für mich okay, wenn jeder andere diese Handlung mit mir als Opfer ausführt.

Der kategorische Imperativ liftet dies eine Abstraktionsebene:

lemma $\langle \text{kategorischer-imperativ wps welt m} \implies$
 $\text{wohlgeformte-handlungsabsicht wps welt ha} \implies$
 $\text{ausfuehrbar ich welt ha} \implies$

$\text{okay } m \text{ ich } (\text{handeln } ich \text{ welt } ha) \implies \text{moralisch } welt \text{ } m \text{ } ha \rangle$

In Worten: Wenn eine Maxime den kategorischen Imperativ erfüllt und es für eine beliebige (wohlgeformte) Handlung auszuführen für mich okay ist diese auszuführen, dann ist diese Handlung moralisch..

Für Beispiele wird es einfacher zu zeigen, dass eine Maxime nicht den kategorischen Imperativ erfüllt, wenn wir direkt ein Beispiel angeben.

definition $\langle \text{kategorischer-imperativ-gegenbeispiel } wps \text{ welt } m \text{ } ha \text{ ich } p1 \text{ } p2 \equiv$
 $\text{wohlgeformte-handlungsabsicht } wps \text{ welt } ha \wedge$
 $\text{ausfuehrbar } ich \text{ welt } ha \wedge \text{okay } m \text{ ich } (\text{handeln } ich \text{ welt } ha) \wedge$
 $\neg \text{okay } m \text{ } p1 \text{ } (\text{handeln } p2 \text{ welt } ha) \rangle$

lemma $\langle \text{kategorischer-imperativ-gegenbeispiel } wps \text{ welt } m \text{ } ha \text{ ich } p1 \text{ } p2 \implies$
 $\neg \text{kategorischer-imperativ } wps \text{ welt } m \rangle$

8.3 Maximen die den Kategorischen Imperativ immer Erfüllen

Wenn eine Maxime jede Handlungsabsicht als moralisch bewertet, erfüllt diese Maxime den kategorischen Imperativ. Da diese Maxime jede Handlung erlaubt, ist es dennoch eine wohl ungeeignete Maxime.

lemma $\langle \forall ha. \text{moralisch } welt \text{ } maxime \text{ } ha \implies \text{kategorischer-imperativ } wps \text{ welt } maxime \rangle$

Eine Maxime die das ich und die Handlung ignoriert erfüllt den kategorischen Imperativ.

lemma *blinde-maxime-katimp*:
 $\langle \text{kategorischer-imperativ } wps \text{ welt } (\text{Maxime } (\lambda ich \text{ } h. \text{ } m)) \rangle$

Eine Maxime welche das *ich* ignoriert, also nur die Handlung global betrachtet, erfüllt den kategorischen Imperativ.

theorem *globale-maxime-katimp*:
fixes $P :: \langle 'world \text{ } handlung \implies bool \rangle$
assumes $mhg: \langle \forall p. \text{maxime-und-handlungsabsicht-generalisieren } wps \text{ welt } (\text{Maxime } (\lambda ich :: 'person. P)) \text{ } ha$
 $p \rangle$
and $\text{maxime-erlaubt-untaetigkeit}: \langle \forall p. \text{ist-noop } (\text{handeln } p \text{ welt } ha) \longrightarrow \text{okay } (\text{Maxime } (\lambda ich :: 'person. P))$
 $p \text{ } (\text{handeln } p \text{ welt } ha) \rangle$
and $\text{kom}: \langle \text{wpsm-kommutiert } (\text{Maxime } (\lambda ich :: 'person. P)) \text{ } wps \text{ welt} \rangle$
and wps-sym :
 $\langle \forall p1 \text{ } p2 \text{ welt. } wps \text{ } p1 \text{ } p2 \text{ welt} = wps \text{ } p2 \text{ } p1 \text{ welt} \rangle$
and wps-id :
 $\langle \forall p1 \text{ } p2 \text{ welt. } wps \text{ } p1 \text{ } p2 \text{ } (wps \text{ } p1 \text{ } p2 \text{ welt}) = welt \rangle$
and $\text{wfh}: \langle \text{wohlgeformte-handlungsabsicht } wps \text{ welt } ha \rangle$
shows $\langle \text{kategorischer-imperativ-auf } ha \text{ welt } (\text{Maxime } (\lambda ich :: 'person. P)) \rangle$

9 Ausführbarer Beispielgenerator

value $\langle [(x,y). x \leftarrow xs, y \leftarrow ys, x \neq y] \rangle$

definition *alle-moeglichen-handlungen*

$:: \langle 'world \Rightarrow ('person::enum, 'world) handlungsabsicht list \Rightarrow 'world handlung list \rangle$

where

$\langle alle-moeglichen-handlungen\ welt\ has \equiv [handeln\ p\ welt\ ha. ha \leftarrow has, p \leftarrow (Enum.enum::'person\ list)] \rangle$

lemma *set-alle-moeglichen-handlungen:*

$\langle set\ (alle-moeglichen-handlungen\ welt\ has) = \{handeln\ p\ welt\ ha \mid ha\ p. ha \in set\ has\} \rangle$

record $('person, 'world) beispiel =$

bsp-welt $:: \langle 'world \rangle$

bsp-erfuellte-maxime $:: \langle ('person, 'world) maxime\ option \rangle$

bsp-erlaubte-handlungen $:: \langle ('person, 'world) handlungsabsicht\ list \rangle$

bsp-verbotene-handlungen $:: \langle ('person, 'world) handlungsabsicht\ list \rangle$

definition *erzeuge-beispiel*

$:: \langle ('person::enum, 'world) wp-swap \Rightarrow 'world \Rightarrow$

$('person, 'world) handlungsabsicht\ list \Rightarrow ('person, 'world) maxime$

$\Rightarrow ('person, 'world) beispiel\ option \rangle$

where

$\langle erzeuge-beispiel\ wps\ welt\ has\ m \equiv$

if $(\exists h \in set\ (alle-moeglichen-handlungen\ welt\ has). \neg wohlgeformte-maxime-auf\ h\ wps\ m)$

$\vee (\exists ha \in set\ has. \neg wohlgeformte-handlungsabsicht\ wps\ welt\ ha)$

then *None*

else *Some*

$\langle \mid bsp-welt = welt,$

bsp-erfuellte-maxime $= if\ \forall ha \in set\ has. kategorischer-imperativ-auf\ ha\ welt\ m\ then\ Some\ m\ else\ None,$

bsp-erlaubte-handlungen $= [ha \leftarrow has. moralisch\ welt\ m\ ha],$

bsp-verbotene-handlungen $= [ha \leftarrow has. \neg moralisch\ welt\ m\ ha]$

$\mid \rangle$

erzeuge-beispiel erzeugt nur ein Beiespiel wenn alles wohlgeformt ist.

lemma *erzeuge-beispiel wps welt has m = Some bsp \implies*

$(\forall ha \in set\ has. wohlgeformte-handlungsabsicht\ wps\ welt\ ha) \wedge$

$(\forall h \in set\ (alle-moeglichen-handlungen\ welt\ has). wohlgeformte-maxime-auf\ h\ wps\ m)$

- Wenn *bsp-erfuellte-maxime* einen *Some* term enthält ist der *kategorischer-imperativ-auf* den Handlungen erfüllt
- Die *bsp-erlaubte-handlungen* und *bsp-verbotene-handlungen* entspricht quasi dem allgemeinen Gesetz, welches besagt, welche Handlungen erlaubt oder verboten sind.

```

lemma⟨erzeuge-beispiel swap (λp::person. 0::int) [Handlungsabsicht (λp w. Some w)] (Maxime (λich w.
True))⟩
=
Some
(⟦bsp-welt = (λp::person. 0::int),
  bsp-erfuellte-maxime = Some (Maxime (λich w. True)),
  bsp-erlaubte-handlungen = [Handlungsabsicht (λp w. Some w)],
  bsp-verbotene-handlungen = []
  ⟧⟩

```

Der Nachteil von *erzeuge-beispiel* ist, dass der resultierende Record viele Funktionen enthält, welche eigentlich nicht geprintet werden können. Allerdings ist dies vermutlich die einzige (sinnvolle, einfache) Art eine Handlungsabsicht darzustellen.

Es wäre einfacher, nur die Handlung (also die *'world handlung*, nur die Welt vorher und nachher, ohne Absicht) aufzuschreiben. Allerdings erzeugt das ohne die Absicht (i.e. (*'person*, *'world*) *handlungsabsicht*) sehr viel Unfug, da z.B. pathologische Grenzfälle (wie z.B. sich-selbst-bestehen, oder die-welt-die-zufällig-im-ausgangszustand-ist-resetten) dazu, dass diese no-op Handlungen verboten sind, da die dahinterliegende Absicht schlecht ist. Wenn wir allerdings nur die Ergebnisse einer solchen Handlung (ohne die Absicht) aufschreiben kommt heraus: Nichtstun ist verboten.

9.1 Kombination vom Maximen

9.1.1 Konjunktion

lemma *MaximeConjI*:

kategorischer-imperativ-auf ha welt m1 \wedge *kategorischer-imperativ-auf ha welt m2* \implies
kategorischer-imperativ-auf ha welt (MaximeConj m1 m2)

definition *ex-erfuellbare-instanz*

$:: ('person, 'world) maxime \Rightarrow 'world \Rightarrow ('person, 'world) handlungsabsicht \Rightarrow bool$

where

ex-erfuellbare-instanz m welt ha \equiv

$\exists ich. ausfuehrbar\ ich\ welt\ ha \wedge okay\ m\ ich\ (handeln\ ich\ welt\ ha)$

Die Rückrichtung gilt nur, wenn wir annehmen, dass es auch einen Fall gibt in dem die *MaximeConj* auch erfüllbar ist:

lemma *MaximeConjD*:

ex-erfuellbare-instanz (MaximeConj m1 m2) welt ha \implies

kategorischer-imperativ-auf ha welt (MaximeConj m1 m2) \implies

kategorischer-imperativ-auf ha welt m1 \wedge *kategorischer-imperativ-auf ha welt m2*

lemma *MaximeConj*:

ex-erfuellbare-instanz (*MaximeConj* *m1 m2*) *welt ha* \implies
kategorischer-imperativ-auf ha welt (*MaximeConj* *m1 m2*) \longleftrightarrow
kategorischer-imperativ-auf ha welt *m1* \wedge *kategorischer-imperativ-auf ha welt* *m2*

lemma *kategorischer-imperativ-auf-MaximeConj-comm*:

kategorischer-imperativ-auf ha welt (*MaximeConj* *m1 m2*)
 \longleftrightarrow *kategorischer-imperativ-auf ha welt* (*MaximeConj* *m2 m1*)

lemma *kategorischer-imperativ-auf-MaximeConj-True*:

kategorischer-imperativ-auf ha welt (*MaximeConj* *m1* (*Maxime* (λ - -. *True*)))
 \longleftrightarrow *kategorischer-imperativ-auf ha welt* *m1*

Achtung: Das ist das Gegenteil, was man von einer Konjunktion erwarten würde. Normalerweise ist $a \wedge \text{False} = \text{False}$. Bei *MaximeConj* ist dies aber *True*! Dies liegt daran, dass *Maxime* (λ - -. *False*) keine Handlung erlaubt, und damit als pathologischen Grenzfall den kategorischen Imperativ erfüllt.

lemma *kategorischer-imperativ-auf-MaximeConj-False*:

kategorischer-imperativ-auf ha welt (*MaximeConj* *m1* (*Maxime* (λ - -. *False*)))

9.1.2 Disjunktion

Für *MaximeDisj* müssen wir generell annehmen, dass einer der Fälle erfüllbar ist.

lemma *kategorischer-imperativ-auf-MaximeDisjI*:

(*ex-erfuellbare-instanz* *m1 welt ha* \wedge *kategorischer-imperativ-auf ha welt* *m1*) \vee
(*ex-erfuellbare-instanz* *m2 welt ha* \wedge *kategorischer-imperativ-auf ha welt* *m2*) \implies
kategorischer-imperativ-auf ha welt (*MaximeDisj* *m1 m2*)

Die Rückrichtung gilt leider nicht.

Die Annahmen sind leider sehr stark:

lemma

ex-erfuellbare-instanz *m welt ha* \wedge *kategorischer-imperativ-auf ha welt* *m*
 \implies
moralisch welt m ha

Wenn wir die Annahme stärker machen gilt auch folgendes:

lemma *kategorischer-imperativ-auf-MaximeDisjI-from-conj*:

kategorischer-imperativ-auf ha welt *m1* \wedge *kategorischer-imperativ-auf ha welt* *m2* \implies
kategorischer-imperativ-auf ha welt (*MaximeDisj* *m1 m2*)

Als Introduction rule eignet sich vermutlich folgendes besser, weil es auch erlaubt, dass eine Handlungsabsicht nicht ausführbar ist oder von keiner Maxime erfüllbar ist.

lemma *kategorischer-imperativ-auf-MaximeDisjI2*:

$$\begin{aligned}
& (ex\text{-erfuellbare-instanz } m1 \text{ welt } ha \wedge kategorischer\text{-imperativ-auf } ha \text{ welt } m1) \vee \\
& (ex\text{-erfuellbare-instanz } m2 \text{ welt } ha \wedge kategorischer\text{-imperativ-auf } ha \text{ welt } m2) \vee \\
& (\neg ex\text{-erfuellbare-instanz } (MaximeDisj \ m1 \ m2) \text{ welt } ha) \\
\implies & \\
& kategorischer\text{-imperativ-auf } ha \text{ welt } (MaximeDisj \ m1 \ m2)
\end{aligned}$$

Die vorherige Introduction Rule lässt sich wie folgt erklären. Mindestens eine der *ex-erfuellbare-instanz*Fälle muss immer zutreffen:

lemma

$$\begin{aligned}
& ex\text{-erfuellbare-instanz } m1 \text{ welt } ha \vee \\
& ex\text{-erfuellbare-instanz } m2 \text{ welt } ha \vee \\
& \neg ex\text{-erfuellbare-instanz } (MaximeDisj \ m1 \ m2) \text{ welt } ha
\end{aligned}$$

Wenn wir also mental den *ex-erfuellbare-instanz* Teil ausblenden, dann liest sich obige Introduction Rule wie folgt: *kategorischer-imperativ-auf ha welt m1* \vee *kategorischer-imperativ-auf ha welt m2* \implies *kategorischer-imperativ-auf ha welt (MaximeDisj m1 m2)*. Dies ist genau die Disjunctions Introduction Rule die ich gerne hätte. Die gesamte Regel ist leider leicht komplizierter, da der entsprechende Oder-Fall immer mit dem entsprechenden *ex-erfuellbare-instanz* gepaart auftreten muss.

Eine gewöhnliche Introduction Rule (ohne die *ex-erfuellbare-instanz* Teile) gilt leider nicht.

lemma

$$\begin{aligned}
& ha = \text{Handlungsabsicht } (\lambda p \ w. \text{Some } w) \implies \\
& m1 = \text{Maxime } ((\lambda p \ h. \text{False})(\text{Bob} := \lambda h. \text{True})) \implies \\
& welt = (0::int) \implies \\
& kategorischer\text{-imperativ-auf } ha \text{ welt } m1 \implies \\
& \neg kategorischer\text{-imperativ-auf } ha \text{ welt } (MaximeDisj \ m1 \ m2)
\end{aligned}$$

zumindest gelten folgende Regeln welche einer gewöhnlichen Disjunctions Introduction ähnlich sehen (mit leicht stärkeren Annahmen):

lemma

$$\begin{aligned}
& (ex\text{-erfuellbare-instanz } m1 \text{ welt } ha \wedge kategorischer\text{-imperativ-auf } ha \text{ welt } m1) \\
& \implies kategorischer\text{-imperativ-auf } ha \text{ welt } (MaximeDisj \ m1 \ m2) \\
& moralisch \text{ welt } m1 \ ha \\
& \implies kategorischer\text{-imperativ-auf } ha \text{ welt } (MaximeDisj \ m1 \ m2)
\end{aligned}$$

lemma *moralisch-kategorischer-imperativ-auf-MaximeDisjI*:

$$\begin{aligned}
& moralisch \text{ welt } m1 \ ha \implies \\
& kategorischer\text{-imperativ-auf } ha \text{ welt } (MaximeDisj \ m1 \ m2)
\end{aligned}$$

lemma *kategorischer-imperativ-auf-MaximeDisj-comm*:

$$\begin{aligned}
& kategorischer\text{-imperativ-auf } ha \text{ welt } (MaximeDisj \ m1 \ m2) \\
& \longleftrightarrow kategorischer\text{-imperativ-auf } ha \text{ welt } (MaximeDisj \ m2 \ m1)
\end{aligned}$$

Für die Grenzfälle einer Disjunktion mit *True* und *False* verhält sich *MaximeDisj* wie erwartet.

```

lemma kategorischer-imperativ-auf-MaximeDisj-True:
  kategorischer-imperativ-auf ha welt (MaximeDisj m1 (Maxime (λ- -. True)))
lemma kategorischer-imperativ-auf-MaximeDisj-False:
  kategorischer-imperativ-auf ha welt (MaximeDisj m1 (Maxime (λ- -. False)))
   $\longleftrightarrow$  kategorischer-imperativ-auf ha welt m1

```

```

fun MaximeNot :: ('person, 'welt) maxime  $\Rightarrow$  ('person, 'welt) maxime
  where
  MaximeNot (Maxime m) = Maxime (λp h.  $\neg$  m p h)

```

```

lemma okay-MaximeNot: okay (MaximeNot m) p h  $\longleftrightarrow$   $\neg$  okay m p h

```

```

declare MaximeNot.simps[simp del]

```

```

lemma kategorischer-imperativ-auf-Maxime-DeMorgan:
  kategorischer-imperativ-auf ha welt (MaximeNot (MaximeConj m1 m2))
   $\longleftrightarrow$ 
  kategorischer-imperativ-auf ha welt (MaximeDisj (MaximeNot m1) (MaximeNot m2))

```

```

lemma kategorischer-imperativ-auf-MaximeNot-double:
  kategorischer-imperativ-auf ha welt (MaximeNot (MaximeNot m))
   $\longleftrightarrow$  kategorischer-imperativ-auf ha welt m

```

10 Utilitarismus

Wir betrachten hier primär einen einfachen Handlungsutilitarismus. Frei nach Jeremy Bentham. Sehr frei. Also sehr viel persönliche Auslegung.

Eine Handlung ist genau dann moralisch richtig, wenn sie den aggregierten Gesamtnutzen, d.h. die Summe des Wohlergehens aller Betroffenen, maximiert wird.

```

type-synonym 'world glueck-messen =  $\langle$  'world handlung  $\Rightarrow$  ereal  $\rangle$ 

```

Wir messen Glück im Typen *ereal*, also reelle Zahlen mit ∞ und $-\infty$, so dass auch "den höchsten Preis zahlen" modelliert werden kann.

```

lemma  $\langle$  (λh::ereal handlung. case h of Handlung vor nach  $\Rightarrow$  nach - vor) (Handlung 3 5) = 2  $\rangle$ 
lemma  $\langle$  (λh::ereal handlung. case h of Handlung vor nach  $\Rightarrow$  nach - vor) (Handlung 3  $\infty$ ) =  $\infty$   $\rangle$ 
lemma  $\langle$  (λh::ereal handlung. case h of Handlung vor nach  $\Rightarrow$  nach - vor) (Handlung 3 ( $-\infty$ )) =  $-\infty$   $\rangle$ 

```

definition *moralisch-richtig* :: $\langle 'world\ glueck-messen \Rightarrow 'world\ handlung \Rightarrow bool \rangle$ **where**
 $\langle moralisch-richtig\ glueck-messen\ handlung \equiv (glueck-messen\ handlung) \geq 0 \rangle$

10.1 Goldene Regel und Utilitarismus im Einklang

In diese kleinen Intermezzo werden wir zeigen, wie sich die Gesinnungsethik der goldenen Regel in die Verantwortungsethik des Utilitarismus übersetzen lässt.

definition *goldene-regel-als-gesinnungsethik*
:: $\langle ('person, 'world)\ maxime \Rightarrow ('person, 'world)\ handlungsabsicht \Rightarrow bool \rangle$
where
 $\langle goldene-regel-als-gesinnungsethik\ maxime\ handlungsabsicht \equiv$
 $\forall\ welt. moralisch\ welt\ maxime\ handlungsabsicht \rangle$

definition *utilitarismus-als-verantwortungsethik*
:: $\langle 'world\ glueck-messen \Rightarrow 'world\ handlung \Rightarrow bool \rangle$
where
 $\langle utilitarismus-als-verantwortungsethik\ glueck-messen\ handlung \equiv$
 $moralisch-richtig\ glueck-messen\ handlung \rangle$

Eine Maxime ist immer aus Sicht einer bestimmten Person definiert. Wir "neutralisieren" eine Maxime indem wir diese bestimmte Person entfernen und die Maxime so allgemeingültiger machen. Alle Personen müssen gleich behandelt werden Um die Maxime unabhängig von einer bestimmten Person zu machen, fordern wir einfach, dass die Maxime für aller Personen erfüllt sein muss.

fun *maximeNeutralisieren* :: $\langle ('person, 'world)\ maxime \Rightarrow ('world\ handlung \Rightarrow bool) \rangle$ **where**
 $\langle maximeNeutralisieren\ (Maxime\ m) = (\lambda\ welt. \forall\ p::'person. m\ p\ welt) \rangle$

Nun übersetzen wir eine Maxime in die *'world glueck-messen* Funktion des Utilitarismus. Der Trick: eine verletzte Maxime wird als unendliches Leid übersetzt.

definition *maxime-als-nutzenkalkuel*
:: $\langle ('person, 'world)\ maxime \Rightarrow 'world\ glueck-messen \rangle$
where
 $\langle maxime-als-nutzenkalkuel\ maxime \equiv$
 $(\lambda\ welt. case\ (maximeNeutralisieren\ maxime)\ welt$
 $of\ True \Rightarrow 1$
 $| False \Rightarrow -\infty) \rangle$

Für diese Übersetzung können wir beweisen, dass die Gesinnungsethik der goldenen Regel und die utilitaristische Verantwortungsethik konsistent sind:

theorem *gesinnungsethik-verantwortungsethik-konsistent*
 $(goldene-regel-als-gesinnungsethik\ maxime)$
 $(utilitarismus-als-verantwortungsethik\ (maxime-als-nutzenkalkuel\ maxime)) \rangle$

Diese Konsistenz gilt nicht im allgemeinen, sondern nur wenn Glück gemessen wird mit Hilfe der *maxime-als-nutzenkalkuel* Funktion. Der Trick dabei ist nicht, dass wir einer verletzten Maxime –

∞ Nutzen zuordnen, sondern der Trick besteht in *maximeNeutralisieren*, welche nicht erlaubt Glück aufzuaddieren und mit Leid zu verrechnen, sondern dank des Allquantors dafür sorgt, dass auch nur das kleinste Leid dazu führt, dass sofort *False* zurückgegeben wird.

Aber wenn wir ordentlich aufsummieren, jedoch einer verletzten Maxime $-\infty$ Nutzen zuordnen und zusätzlich annehmen, dass die Bevölkerung endlich ist, dann funktioniert das auch:

```
fun maxime-als-summe-wohlergehen
  :: ⟨('person, 'world) maxime  $\Rightarrow$  'world glueck-messen⟩
where
  ⟨maxime-als-summe-wohlergehen (Maxime m) =
    (λwelt.  $\sum_{p \in \text{bevoelkerung.}}$  (case m p welt
      of True  $\Rightarrow$  1
      | False  $\Rightarrow -\infty$ ))⟩

theorem
  fixes maxime :: ⟨('person, 'world) maxime⟩
  assumes ⟨finite (bevoelkerung:: 'person set)⟩
  shows
    ⟨gesinnungsethik-verantwortungsethik-konsistent
      (goldene-regel-als-gesinnungsethik maxime)
      (utilitarismus-als-verantwortungsethik (maxime-als-summe-wohlergehen maxime))⟩
```

"Wie zu erwarten, will Kant nichts vom Utilitarismus oder sonstigen Lehren wissen, die der Moral einen außerhalb ihrer selbst liegenden Zweck zuschreiben" [1]. Die eben bewiesene Konsistenz von Gesinnungsethik und Verantwortungsethik zeigt, dass unsere Grunddefinition bereits eine Formalisierung des Kategorischen Imperativs komplett im strengen Sinne Kants ausschließen. Dennoch finde ich unsere Interpretation bis jetzt nicht abwegig.

11 Zahlenwelt Helper

Wir werden Beispiele betrachten, in denen wir Welten modellieren, in denen jeder Person eine Zahl zugewiesen wird: *person* \Rightarrow *int*. Diese Zahl kann zum Beispiel der Besitz oder Wohlstand einer Person sein, oder das Einkommen. Wobei Gesamtbesitz und Einkommen über einen kurzen Zeitraum recht unterschiedliche Sachen modellieren.

Hier sind einige Hilfsfunktionen um mit *person* \Rightarrow *int* allgemein zu arbeiten.


Default: Standardmäßig hat jede Person 0:

```
definition DEFAULT :: ⟨person  $\Rightarrow$  int⟩ where
  ⟨DEFAULT  $\equiv \lambda p. 0$ ⟩
```

Beispiel:

```
lemma ⟨(DEFAULT(Alice:=8, Bob:=3, Eve:= 5)) Bob = 3⟩
```

Beispiel mit fancy Syntax:

```
lemma ⟨[Alice:=8, Bob:=3, Eve:= 5] Bob = 3⟩
```

lemma $\langle \text{show-fun } \clubsuit[Alice := 4, Carol := 4] = [(Alice, 4), (Bob, 0), (Carol, 4), (Eve, 0)] \rangle$

lemma $\langle \text{show-num-fun } \clubsuit[Alice := 4, Carol := 4] = [(Alice, 4), (Carol, 4)] \rangle$

abbreviation *num-fun-add-syntax* ($\llbracket \cdot \text{ '(- += -)'} \rrbracket$) **where**

$\langle \llbracket f(p += n) \rrbracket \equiv (f(p := (f\ p) + n)) \rangle$

abbreviation *num-fun-minus-syntax* ($\llbracket \cdot \text{ '(- -= -)'} \rrbracket$) **where**

$\langle \llbracket f(p -= n) \rrbracket \equiv (f(p := (f\ p) - n)) \rangle$

lemma $\langle \llbracket \clubsuit[Alice:=8, Bob:=3, Eve:= 5](Bob += 4) \rrbracket\ Bob = 7 \rangle$

lemma $\langle \llbracket \clubsuit[Alice:=8, Bob:=3, Eve:= 5](Bob -= 4) \rrbracket\ Bob = -1 \rangle$

lemma *fixes* $n :: \langle \text{int} \rangle$ **shows** $\langle \llbracket f(p += n) \rrbracket(p -= n) \rrbracket = f \rangle$

Diskriminierungsfrei eine *'person* eindeutig anhand Ihres Besitzes auswählen:

definition *opfer-eindeutig-nach-besitz-auswaehlen*

$:: \langle \text{int} \Rightarrow ('person \Rightarrow \text{int}) \Rightarrow 'person\ list \Rightarrow 'person\ option \rangle$ **where**

$\langle \text{opfer-eindeutig-nach-besitz-auswaehlen } b\ \text{besitz } ps =$

$(\text{case filter } (\lambda p. \text{besitz } p = b)\ ps$

$\text{of } [opfer] \Rightarrow \text{Some } opfer$

$| _ \Rightarrow \text{None} \rangle$

definition *the-single-elem* $:: \langle 'a\ set \Rightarrow 'a\ option \rangle$ **where**

$\langle \text{the-single-elem } s \equiv \text{if card } s = 1 \text{ then Some (Set.the-elem } s) \text{ else None} \rangle$

thm *is-singleton-the-elem* $[\text{symmetric}]$

lemma $\langle A = \{\text{the-elem } A\} \longleftrightarrow \text{is-singleton } A \rangle$

lemma *opfer-nach-besitz-induct-step-set-simp*: $\langle \text{besitz } a \neq \text{opfer-nach-besitz} \implies$

$\{p. (p = a \vee p \in \text{set } ps) \wedge \text{besitz } p = \text{opfer-nach-besitz}\} =$

$\{p \in \text{set } ps. \text{besitz } p = \text{opfer-nach-besitz}\} \rangle$

lemma *opfer-eindeutig-nach-besitz-auswaehlen-the-single-elem*:

$\langle \text{distinct } ps \implies$

$\text{opfer-eindeutig-nach-besitz-auswaehlen } \text{opfer-nach-besitz } \text{besitz } ps =$

$\text{the-single-elem } \{p \in \text{set } ps. \text{besitz } p = \text{opfer-nach-besitz}\} \rangle$

lemma *opfer-eindeutig-nach-besitz-auswaehlen-the-single-elem-enumall*:

$\langle \text{opfer-eindeutig-nach-besitz-auswaehlen } \text{opfer-nach-besitz } \text{besitz } \text{enum-class.enum} =$

$\text{the-single-elem } \{p. \text{besitz } p = \text{opfer-nach-besitz}\} \rangle$

fun *stehlen* $:: \langle \text{int} \Rightarrow \text{int} \Rightarrow 'person :: \text{enum} \Rightarrow ('person \Rightarrow \text{int}) \Rightarrow ('person \Rightarrow \text{int})\ option \rangle$ **where**

$\langle \text{stehlen } \text{beute } \text{opfer-nach-besitz } \text{dieb } \text{besitz} =$

$(\text{case } \text{opfer-eindeutig-nach-besitz-auswaehlen } \text{opfer-nach-besitz } \text{besitz } \text{Enum.enum}$

```

of None  $\Rightarrow$  None
| Some opfer  $\Rightarrow$  if opfer = dieb then None else Some  $\llbracket$ besitz(opfer  $\text{--}$  beute) $\rrbracket$ (dieb  $\text{+}$  beute) $\rrbracket$ 
) $\rangle$ 

```

lemma wohlgeformte-handlungsabsicht-stehlen:
 \langle wohlgeformte-handlungsabsicht swap welt (Handlungsabsicht (stehlen n p)) \rangle

definition aufsummieren :: \langle ('person::enum \Rightarrow int) \Rightarrow int \rangle **where**
 \langle aufsummieren besitz = sum-list (map besitz Enum.enum) \rangle

lemma \langle aufsummieren (besitz :: person \Rightarrow int) = ($\sum p \leftarrow$ [Alice,Bob,Carol,Eve]. besitz p) \rangle

lemma \langle aufsummieren \clubsuit [Alice := 4, Carol := 8] = 12 \rangle

lemma \langle aufsummieren \clubsuit [Alice := 4, Carol := 4] = 8 \rangle

lemma aufsummieren-swap:
 \langle aufsummieren (swap p1 p2 welt) = aufsummieren welt \rangle

lemma list-not-empty-iff-has-element: $as \neq [] \longleftrightarrow (\exists a. a \in \text{set } as)$

lemma enum-class-not-empty-list: $\text{enum-class.enum} \neq []$

lemma alles-kaputt-machen-code-help:
 $(\lambda-. \text{Min } (\text{range } x) - 1) = (\lambda-. \text{min-list } (\text{map } x \text{ enum-class.enum}) - 1)$

12 Beispiel: Zahlenwelt

Wir nehmen an, die Welt lässt sich durch eine Zahl darstellen, die den Besitz einer Person modelliert. Der Besitz ist als ganze Zahl *int* modelliert und kann auch beliebig negativ werden.

datatype zahlenwelt = Zahlenwelt
 \langle person \Rightarrow int — besitz: Besitz jeder Person. \rangle

fun gesamtbesitz :: \langle zahlenwelt \Rightarrow int \rangle **where**
 \langle gesamtbesitz (Zahlenwelt besitz) = aufsummieren besitz \rangle

Beispiel:

```

lemma <gesamtbesitz (Zahlenwelt ♣[Alice := 4, Carol := 8]) = 12>
lemma <gesamtbesitz (Zahlenwelt ♣[Alice := 4, Carol := 4]) = 8>

```

Mein persönlicher Besitz:

```

fun meus :: <person ⇒ zahlenwelt ⇒ int> where
  <meus p (Zahlenwelt besitz) = besitz p>

```

Beispiel:

```

lemma <meus Carol (Zahlenwelt ♣[Alice := 8, Carol := 4]) = 4>

```

Um den `SchleierNichtwissen.thy` zu implementieren:

```

fun zahlenwps :: <person ⇒ person ⇒ zahlenwelt ⇒ zahlenwelt> where
  <zahlenwps p1 p2 (Zahlenwelt besitz) = Zahlenwelt (swap p1 p2 besitz)>

```

Beispiel:

```

lemma <zahlenwps Alice Carol (Zahlenwelt ♣[Alice := 4, Bob := 6, Carol := 8])
  = (Zahlenwelt ♣[Alice := 8, Bob := 6, Carol := 4])>

```

Alice hat Besitz, Bob ist reicher, Carol hat Schulden.

```

definition <initialwelt ≡ Zahlenwelt ♣[Alice := 5, Bob := 10, Carol := -3]>

```

12.1 Ungültige Handlung

Sobald ich eine konkrete Person in einer Handlungsabsicht hardcode, ist diese nicht mehr wohlgeformt.

```

lemma <¬wohlgeformte-handlungsabsicht
  zahlenwps (Zahlenwelt ♣[Alice := 5, Bob := 10, Carol := -3])
  (Handlungsabsicht (λich w. if ich = Alice then Some w else Some (Zahlenwelt (λ-. 0))))>

```

12.2 Nicht-Wohlgeformte Handlungen

```

fun stehlen-nichtwf :: <int ⇒ person ⇒ person ⇒ zahlenwelt ⇒ zahlenwelt option> where
  <stehlen-nichtwf beute opfer dieb (Zahlenwelt besitz) =
    Some (Zahlenwelt ([[besitz(opfer -= beute)]](dieb += beute))]>

```

Die Handlung `stehlen` diskriminiert und ist damit nicht wohlgeformt:

```

lemma <wohlgeformte-handlungsabsicht-gegenbeispiel zahlenwps
  (Zahlenwelt (λx. 0)) (Handlungsabsicht (stehlen-nichtwf 5 Bob))
  Alice Bob>

```

Wir versuchen, das Opfer nach Besitz auszuwählen, nicht nach Namen. Nach unserer Definition ist der Besitz ein Merkmal, nach dem man diskriminieren darf. Man darf nur nicht nach Eigenschaften der `person` diskriminieren, sondern nur nach Eigenschaften der `zahlenwelt`.

```

fun opfer-nach-besitz-auswaehlen :: <int ⇒ ('person ⇒ int) ⇒ 'person list ⇒ 'person option> where
  <opfer-nach-besitz-auswaehlen - - [] = None>

```



```

| <opfer-nach-besitz-auswaehlen b besitz (p#ps) =
  (if besitz p = b then Some p else opfer-nach-besitz-auswaehlen b besitz ps)>

fun stehlen-nichtwf2 :: <int ⇒ int ⇒ person ⇒ zahlenwelt ⇒ zahlenwelt option> where
  <stehlen-nichtwf2 beute opfer-nach-besitz dieb (Zahlenwelt besitz) =
    (case opfer-nach-besitz-auswaehlen opfer-nach-besitz besitz Enum.enum
      of None ⇒ None
       | Some opfer ⇒ Some (Zahlenwelt [[besitz(opfer -= beute)](dieb += beute)]))
  >

```

Leider ist diese Funktion auch diskriminierend: Wenn es mehrere potenzielle Opfer mit dem gleichen Besitz gibt, dann bestimmt die Reihenfolge in *enum-class.enum* wer bestohlen wird. Diese Reihenfolge ist wieder eine Eigenschaft von *person* und nicht *zahlenwelt*.

```

lemma <handeln Alice (Zahlenwelt ♣[Alice := 10, Bob := 10, Carol := -3])
  (Handlungsabsicht (stehlen-nichtwf2 5 10))
  = Handlung (Zahlenwelt ♣[Alice := 10, Bob := 10, Carol := -3])
  (Zahlenwelt ♣[Alice := 10, Bob := 10, Carol := -3])>
lemma <handeln Bob (Zahlenwelt ♣[Alice := 10, Bob := 10, Carol := -3])
  (Handlungsabsicht (stehlen-nichtwf2 5 10))
  = Handlung (Zahlenwelt ♣[Alice := 10, Bob := 10, Carol := -3])
  (Zahlenwelt ♣[Alice := 5, Bob := 15, Carol := -3])>
lemma <wohlgeformte-handlungsabsicht-gegenbeispiel
  zahlenwps
  (Zahlenwelt ♣[Alice := 10, Bob := 10, Carol := -3]) (Handlungsabsicht (stehlen-nichtwf2 5 10))
  Alice Bob>

```

```

fun schenken :: <int ⇒ person ⇒ person ⇒ zahlenwelt ⇒ zahlenwelt option> where
  <schenken betrag empfaenger schenker (Zahlenwelt besitz) =
    Some (Zahlenwelt [[besitz(schenker -= betrag)](empfaenger += betrag)])>

```

Da wir ganze Zahlen verwenden und der Besitz auch beliebig negativ werden kann, ist Stehlen äquivalent dazu einen negativen Betrag zu verschenken:

```

lemma stehlen-ist-schenken: <stehlen-nichtwf i = schenken (-i)>

```

Das Modell ist nicht ganz perfekt, Aber passt schon um damit zu spielen.

12.3 Wohlgeformte Handlungen

Die folgende Handlung erschafft neuen Besitz aus dem Nichts:

```

fun erschaffen :: <nat ⇒ person ⇒ zahlenwelt ⇒ zahlenwelt option> where
  <erschaffen i p (Zahlenwelt besitz) = Some (Zahlenwelt [[besitz(p += int i)])>
lemma <wohlgeformte-handlungsabsicht zahlenwps welt (Handlungsabsicht (erschaffen n))>

```

Wenn wir das Opfer eindeutig auswählen, ist die Handlung wohlgeformt. Allerdings wird niemand bestohlen, wenn das Opfer nicht eindeutig ist.

```
fun stehlen4 :: ⟨int ⇒ int ⇒ person ⇒ zahlenwelt ⇒ zahlenwelt option⟩ where
  ⟨stehlen4 beute opfer-nach-besitz dieb (Zahlenwelt besitz) =
    map-option Zahlenwelt (stehlen beute opfer-nach-besitz dieb besitz)⟩
```

Reset versetzt die Welt wieder in den Ausgangszustand. Eine sehr destruktive Handlung.

```
fun reset :: ⟨person ⇒ zahlenwelt ⇒ zahlenwelt option⟩ where
  ⟨reset ich (Zahlenwelt besitz) = Some (Zahlenwelt (λ -. 0))⟩
```

Der *reset* ist im moralischen Sinne vermutlich keine gute Handlung, dennoch ist es eine wohlgeformte Handlung, welche wir betrachten können:

```
lemma ⟨wohlgeformte-handlungsabsicht zahlenwps welt (Handlungsabsicht reset)⟩
```

```
fun alles-kaputt-machen :: ⟨person ⇒ zahlenwelt ⇒ zahlenwelt option⟩ where
  ⟨alles-kaputt-machen ich (Zahlenwelt besitz) = Some (Zahlenwelt (λ -. Min (besitz ‘ UNIV) - 1))⟩
```

```
lemma ⟨alles-kaputt-machen Alice (Zahlenwelt ♣[Alice := 5, Bob := 10, Carol := -3])
  = Some (Zahlenwelt ♣[Alice := -4, Bob := -4, Carol := -4, Eve := -4])⟩
```

```
fun unmoeiglich :: ⟨person ⇒ zahlenwelt ⇒ zahlenwelt option⟩ where
  ⟨unmoeiglich - - = None⟩
```

Die Beispielhandlungsabsichten, die wir betrachten wollen.

```
definition handlungsabsichten ≡ [
  Handlungsabsicht (erschaffen 5),
  Handlungsabsicht (stehlen4 5 10),
  Handlungsabsicht reset,
  Handlungsabsicht alles-kaputt-machen,
  Handlungsabsicht unmoeiglich
]
```

```
lemma ⟨ha ∈ set handlungsabsichten ⇒ wohlgeformte-handlungsabsicht zahlenwps welt ha⟩
```

12.4 Maxime für individuellen Fortschritt

Wir definieren eine Maxime die besagt, dass sich der Besitz einer Person nicht verringern darf:

```
fun individueller-fortschritt :: ⟨person ⇒ zahlenwelt handlung ⇒ bool⟩ where
  ⟨individueller-fortschritt p (Handlung vor nach) ⇔ (meins p vor) ≤ (meins p nach)⟩
```

```
definition maxime-zahlenfortschritt :: ⟨(person, zahlenwelt) maxime⟩ where
```

$\langle \text{maxime-zahlenfortschritt} \equiv \text{Maxime} (\lambda \text{ich. individueller-fortschritt ich}) \rangle$

reset erfüllt das nicht, aber das normale *stehlen*.

lemma $ha \in \{$
Handlungsabsicht (erschaffen 5),
Handlungsabsicht (stehlen-nichtwf 5 Bob),
Handlungsabsicht (stehlen4 5 10),
Handlungsabsicht alles-kaputt-machen,
Handlungsabsicht unmöglich
 $\} \implies \text{maxime-und-handlungsabsicht-generalisieren zahlenwps welt maxime-zahlenfortschritt ha p}$

Nicht alle Handlungen generalisieren, z.B. *reset* nicht:

lemma
 $\langle \neg \text{maxime-und-handlungsabsicht-generalisieren}$
 $\text{zahlenwps (Zahlenwelt } \clubsuit [\text{Alice} := 5, \text{Bob} := 10, \text{Carol} := -3])$
 $\text{maxime-zahlenfortschritt (Handlungsabsicht (reset)) Alice} \rangle$

Die *maxime-zahlenfortschritt* erfüllt **nicht** den *kategorischer-imperativ* da *Alice* nach der *Maxime* z.B. *Bob* bestehlen dürfte.

lemma $\langle \text{kategorischer-imperativ-gegenbeispiel}$
 $\text{zahlenwps initialwelt maxime-zahlenfortschritt}$
 $(\text{Handlungsabsicht (stehlen4 1 10)})$
 Alice
 Bob
 $\text{Alice} \rangle$

12.4.1 Einzellbeispiele

In jeder Welt ist die *Handlungsabsicht (erschaffen n)* *moralisch*:

lemma $\langle \text{moralisch welt maxime-zahlenfortschritt (Handlungsabsicht (erschaffen n))} \rangle$

In kein Welt ist *Stehlen* *moralisch*:

lemma $\langle \neg \text{moralisch welt maxime-zahlenfortschritt (Handlungsabsicht (stehlen-nichtwf 5 Bob))} \rangle$

In unserer *initialwelt* in der *Bob* als Opfer anhand seines Besitzes als Opfer eines Diebstahls ausgewählt würde, ist *stehlen* dennoch nicht *moralisch*, obwohl die *Handlungsabsicht* wohlgeformt ist:

lemma $\langle \neg \text{moralisch initialwelt maxime-zahlenfortschritt (Handlungsabsicht (stehlen4 5 10))} \rangle$

Da Schenken und Stehlen in dieser Welt equivalent ist, ist Schenken auch unmoralisch:

lemma $\langle \neg \text{moralisch welt maxime-zahlenfortschritt (Handlungsabsicht (schenken 5 Bob))} \rangle$

TODO: erklären

```

lemma <erzeuge-beispiel
  zahlenwps initialwelt
  handlungsabsichten
  (Maxime individueller-fortschritt) =
Some
  (bsp-welt = Zahlenwelt ♣[Alice := 5, Bob := 10, Carol := -3],
   bsp-erfuellte-maxime = None,
   bsp-erlaubte-handlungen = [Handlungsabsicht (erschaffen 5), Handlungsabsicht unmoeiglich],
   bsp-verbotene-handlungen = [Handlungsabsicht (stehlen4 5 10), Handlungsabsicht reset, Handlungsabsicht
alles-kaputt-machen])>

```

12.5 Maxime für allgemeinen Fortschritt

Allerdings können wir die Maxime generalisieren, indem wir *individueller-fortschritt* für jeden fordern. Effektiv wird dabei das *ich* ignoriert.

```

definition maxime-altruistischer-fortschritt :: <(person, zahlenwelt) maxime> where
  <maxime-altruistischer-fortschritt ≡ Maxime (λich h. ∀ pX. individueller-fortschritt pX h)>

```

Folgendes Beispiel zeigt, dass die *maxime-altruistischer-fortschritt* den kategorischen Imperativ (für diese *initialwelt* und *handlungsabsichten*) erfüllt; zu sehen an dem *Some* Term im *bsp-erfuellte-maxime*. Die Handlungsabsichten werden eingeordnet wie erwartet: *erschaffen* ist gut, *stehlen4*, *reset*, *alles-kaputt-machen* ist schlecht.

```

lemma <erzeuge-beispiel
  zahlenwps initialwelt
  handlungsabsichten
  maxime-altruistischer-fortschritt =
Some
  (bsp-welt = Zahlenwelt ♣[Alice := 5, Bob := 10, Carol := -3],
   bsp-erfuellte-maxime = Some maxime-altruistischer-fortschritt,
   bsp-erlaubte-handlungen = [Handlungsabsicht (erschaffen 5), Handlungsabsicht unmoeiglich],
   bsp-verbotene-handlungen = [Handlungsabsicht (stehlen4 5 10), Handlungsabsicht reset, Handlungsabsicht
alles-kaputt-machen])>

```

Das ist ein sehr schönes Beispiel.

Die Aussage, dass die *maxime-altruistischer-fortschritt* den kategorischen Imperativ für bestimmte Handlungsabsichten und Welten erfüllt generalisiert noch weiter. Für alle Welten und alle wohlgeformten Handlungsabsichten welche mit der Maxime generalisieren erfüllt die Maxime den kategorischen Imperativ.

```

theorem <
  ∀ p. maxime-und-handlungsabsicht-generalisieren zahlenwps welt maxime-altruistischer-fortschritt ha p ⇒
  wohlgeformte-handlungsabsicht zahlenwps welt ha ⇒
  kategorischer-imperativ-auf ha welt maxime-altruistischer-fortschritt>

```

Allgemein scheint dies eine sehr gute Maxime zu sein (für dieses sehr beschränkte Weltenmodell).

12.6 Maxime für strikten individuellen Fortschritt

In der Maxime *individueller-fortschritt* hatten wir $\text{meins } p \text{ vor} \leq \text{meins } p \text{ nach}$. Was wenn wir nun echten Fortschritt fordern: $\text{meins } p \text{ vor} < \text{meins } p \text{ nach}$.

```
fun individueller-strikter-fortschritt ::  $\langle \text{person} \Rightarrow \text{zahlenwelt handlung} \Rightarrow \text{bool} \rangle$  where
   $\langle \text{individueller-strikter-fortschritt } p \text{ (Handlung vor nach)} \longleftrightarrow (\text{meins } p \text{ vor}) < (\text{meins } p \text{ nach}) \rangle$ 
```

TODO: erklären. Erfüllt nicht kategorischen imperativ und alles ist verboten

```
lemma  $\langle \text{erzeuge-beispiel}$ 
  zahlenwps initialwelt
  handlungsabsichten
  (Maxime individueller-strikter-fortschritt) =
Some
  ( $\langle \text{bsp-welt} = \text{Zahlenwelt } \clubsuit [\text{Alice} := 5, \text{Bob} := 10, \text{Carol} := -3],$ 
     $\text{bsp-erfuellte-maxime} = \text{None},$ 
     $\text{bsp-erlaubte-handlungen} = [],$ 
     $\text{bsp-verbotene-handlungen} = \text{handlungsabsichten} \rangle$ )
```

In keiner Welt ist die Handlung *erschaffen* nun *moralisch*:

```
lemma  $\langle \neg \text{moralisch welt}$ 
  (Maxime ( $\lambda \text{ich. individueller-strikter-fortschritt ich}$ )) (Handlungsabsicht (erschaffen 5))
```

Der Grund ist, dass der Rest der Bevölkerung keine *strikte* Erhöhung des eigenen Wohlstands erlebt. Effektiv führt diese Maxime zu einem Gesetz, welches es einem Individuum nicht erlaubt mehr Besitz zu erschaffen, obwohl niemand dadurch einen Nachteil hat. Diese Maxime kann meiner Meinung nach nicht gewollt sein.

Beispielsweise ist *Bob* das Opfer wenn *Alice* sich 5 Wohlstand erschafft, aber *Bob's* Wohlstand sich nicht erhöht:

```
lemma  $\langle \text{VerletzteMaxime (Opfer Bob) (Taeter Alice)}$ 
  (Handlung [(Alice, 5), (Bob, 10), (Carol, -3)] [(Alice, 10), (Bob, 10), (Carol, -3)])
   $\in \text{debug-maxime show-zahlenwelt initialwelt}$ 
  (Maxime ( $\lambda \text{ich. individueller-strikter-fortschritt ich}$ )) (Handlungsabsicht (erschaffen 5))
```

12.7 Maxime für globales striktes Optimum

Wir bauen nun eine Maxime, die das Individuum vernachlässigt und nur nach dem globalen Optimum strebt:

```
fun globaler-strikter-fortschritt ::  $\langle \text{zahlenwelt handlung} \Rightarrow \text{bool} \rangle$  where
   $\langle \text{globaler-strikter-fortschritt (Handlung vor nach)} \longleftrightarrow (\text{gesamtbesitz vor}) < (\text{gesamtbesitz nach}) \rangle$ 
```

Die Maxime ignoriert das *ich* komplett.

Nun ist es *Alice* wieder erlaubt, Wohlstand für sich selbst zu erzeugen, da sich dadurch auch der Gesamtwohlstand erhöht:

lemma $\langle \text{moralisch initialwelt}$
 $(\text{Maxime } (\lambda \text{ich. globaler-strikter-fortschritt})) (\text{Handlungsabsicht } (\text{erschaffen } 5)) \rangle$

Allerdings ist auch diese Maxime auch sehr grausam, da sie Untätigkeit verbietet:

lemma $\langle \neg \text{moralisch initialwelt}$
 $(\text{Maxime } (\lambda \text{ich. globaler-strikter-fortschritt})) (\text{Handlungsabsicht } (\text{erschaffen } 0)) \rangle$

Unsere initiale einfache *maxime-zahlenfortschritt* würde Untätigkeit hier erlauben:

lemma $\langle \text{moralisch initialwelt}$
 $\text{maxime-zahlenfortschritt } (\text{Handlungsabsicht } (\text{erschaffen } 0)) \rangle$

TODO: erklären.

lemma $\langle \text{erzeuge-beispiel}$
 $\text{zahlenwps initialwelt}$
 $\text{handlungsabsichten}$
 $(\text{Maxime } (\lambda \text{ich. globaler-strikter-fortschritt})) =$
 Some
 $(\text{bsp-welt} = \text{Zahlenwelt } \clubsuit [\text{Alice} := 5, \text{Bob} := 10, \text{Carol} := -3],$
 $\text{bsp-erfuellte-maxime} = \text{Some } (\text{Maxime } (\lambda \text{ich. globaler-strikter-fortschritt})),$
 $\text{bsp-erlaubte-handlungen} = [\text{Handlungsabsicht } (\text{erschaffen } 5)],$
 $\text{bsp-verbotene-handlungen} = [$
 $\text{Handlungsabsicht } (\text{stehlen4 } 5 \ 10),$
 $\text{Handlungsabsicht reset},$
 $\text{Handlungsabsicht alles-kaputt-machen},$
 $\text{Handlungsabsicht unmöglich}] \rangle$

12.8 Maxime für globales Optimum

Wir können die Maxime für globalen Fortschritt etwas lockern:

fun *globaler-fortschritt* :: $\langle \text{zahlenwelt handlung} \Rightarrow \text{bool} \rangle$ **where**
 $\langle \text{globaler-fortschritt } (\text{Handlung vor nach}) \longleftrightarrow (\text{gesamtbesitz vor}) \leq (\text{gesamtbesitz nach}) \rangle$

Untätigkeit ist nun auch hier erlaubt:

lemma $\langle \text{moralisch initialwelt}$
 $(\text{Maxime } (\lambda \text{ich. globaler-fortschritt})) (\text{Handlungsabsicht } (\text{erschaffen } 0)) \rangle$

theorem

$\langle \forall p. \text{maxime-und-handlungsabsicht-generalisieren zahlenwps welt}$
 $(\text{Maxime } (\lambda \text{ich. globaler-fortschritt})) \text{ ha } p \implies$
 $\text{wohlgeformte-handlungsabsicht zahlenwps welt ha} \implies$
 $\text{kategorischer-imperativ-auf ha welt } (\text{Maxime } (\lambda \text{ich}::\text{person. globaler-fortschritt})) \rangle$

Allerdings ist auch Stehlen erlaubt, da global gesehen, kein Besitz vernichtet wird:

```
lemma <moralisch initialwelt
  (Maxime ( $\lambda$ ich. globaler-fortschritt)) (Handlungsabsicht (stehlen-nichtwf 5 Bob))>
lemma <moralisch initialwelt
  (Maxime ( $\lambda$ ich. globaler-fortschritt)) (Handlungsabsicht (stehlen4 5 10))>
```

TODO: erklären.

```
lemma <erzeuge-beispiel
  zahlenwps initialwelt
  handlungsabsichten
  (Maxime ( $\lambda$ ich. globaler-fortschritt)) =
Some
  (bsp-welt = Zahlenwelt [Alice := 5, Bob := 10, Carol := -3],
   bsp-erfuellte-maxime = Some (Maxime ( $\lambda$ ich. globaler-fortschritt)),
   bsp-erlaubte-handlungen = [
     Handlungsabsicht (erschaffen 5),
     Handlungsabsicht (stehlen4 5 10),
     Handlungsabsicht unmöglich],
   bsp-verbotene-handlungen = [
     Handlungsabsicht reset,
     Handlungsabsicht alles-kaputt-machen])>
```

12.9 Ungültige Maxime

Es ist verboten, in einer Maxime eine spezielle Person hardzucoden. Da dies gegen die Gleichbehandlung aller Menschen verstoßen würde.

Beispielsweise könnten wir *individueller-fortschritt* nicht mehr parametrisiert verwenden, sondern einfach *Alice* reinschreiben:

```
lemma <individueller-fortschritt Alice
  = ( $\lambda$ h. case h of Handlung vor nach  $\Rightarrow$  (meins Alice vor)  $\leq$  (meins Alice nach))>
lemma  $\neg$ wohlgeformte-maxime-auf
  (handeln Alice initialwelt (Handlungsabsicht (stehlen4 5 10))) zahlenwps
  (Maxime ( $\lambda$ ich. individueller-fortschritt Alice))
lemma wohlgeformte-maxime-auf
  (handeln Alice initialwelt (Handlungsabsicht (stehlen4 5 10))) zahlenwps
  (Maxime ( $\lambda$ ich. individueller-fortschritt ich))
```

13 Beispiel: Zahlenwelt2

Konsens laut https://de.wikipedia.org/wiki/Konsens#Konsens_im_Rechtssystem: "die Übereinstimmung der Willenserklärungen beider Vertragspartner über die Punkte des Vertrages"

```
record zahlenwelt =
  besitz :: <person  $\Rightarrow$  int>
```

konsens :: $\langle \text{person} \Rightarrow (\text{person}, \text{int}) \text{ abmachung list} \rangle$
staatsbesitz :: $\langle \text{int} \rangle$ — Der Staat ist keine natürliche Person und damit besonders.
umwelt :: $\langle \text{int} \rangle$

definition *initialwelt* :: *zahlenwelt*

where

initialwelt \equiv \langle
besitz = \clubsuit [*Alice* := 5, *Bob* := 10, *Carol* := -3],
konsens = $(\lambda\cdot. \langle \rangle)$ (
Alice := [aenderung-map [Gewinnt *Alice* 3], aenderung-map [Gewinnt *Alice* 3, Verliert *Bob* 3]],
Bob := [aenderung-map [Gewinnt *Alice* 3, Verliert *Bob* 3]],
staatsbesitz = 9000,
umwelt = 600
 \rangle

Mein persönlicher Besitz:

fun *meins* :: $\langle \text{person} \Rightarrow \text{zahlenwelt} \Rightarrow \text{int} \rangle$ **where**
 $\langle \text{meins } p \text{ welt} = (\text{besitz } \text{welt}) \text{ } p \rangle$

lemma $\langle \text{meins } \text{Carol } \text{initialwelt} = -3 \rangle$

definition *konsensswap*

:: $\langle \text{'person} \Rightarrow \text{'person} \Rightarrow (\text{'person} \Rightarrow (\text{'person}, \text{'etwas}) \text{ abmachung list})$
 $\Rightarrow (\text{'person} \Rightarrow (\text{'person}, \text{'etwas}) \text{ abmachung list})$

where

konsensswap *p1 p2 kons* \equiv *swap* *p1 p2* ((*map* (*swap* *p1 p2*)) \circ *kons*)

lemma *konsensswap-id[simp]*: *konsensswap* *p1 p2* (*konsensswap* *p1 p2 kons*) = *kons*

lemma *konsensswap-sym*: *konsensswap* *p1 p2* = *konsensswap* *p2 p1*

definition *zahlenwps* :: $\langle \text{person} \Rightarrow \text{person} \Rightarrow \text{zahlenwelt} \Rightarrow \text{zahlenwelt} \rangle$ **where**

$\langle \text{zahlenwps } p1 \text{ } p2 \text{ } \text{welt} =$
welt \langle *besitz* := *swap* *p1 p2* (*besitz* *welt*),
konsens := *konsensswap* *p1 p2* (*konsens* *welt*) \rangle

lemma $\langle \text{zahlenwps } \text{Alice } \text{Bob } \text{initialwelt}$

= \langle

besitz = \clubsuit [*Alice* := 10, *Bob* := 5, *Carol* := -3],
konsens = $(\lambda\cdot. \langle \rangle)$ (
Alice := [aenderung-map [Gewinnt *Bob* 3, Verliert *Alice* 3]],
Bob := [aenderung-map [Gewinnt *Bob* 3], aenderung-map [Gewinnt *Bob* 3, Verliert *Alice* 3]],


```

    staatsbesitz = 9000,
    umwelt = 600

```

```

⟩

```

lemma $\langle \text{zahlenwps Alice Carol initialwelt}$

```

= ()

```

```

    besitz = ♣[Alice := -3, Bob := 10, Carol := 5],

```

```

    konsens = (λ-. []) (

```

```

        Bob := [aenderung-map [Gewinnt Carol 3, Verliert Bob 3]],

```

```

        Carol := [aenderung-map [Gewinnt Carol 3], aenderung-map [Gewinnt Carol 3, Verliert Bob 3]],

```

```

    staatsbesitz = 9000,

```

```

    umwelt = 600

```

```

⟩

```

lemma $\text{zahlenwps-id: zahlenwps } p1 \ p2 \ (\text{zahlenwps } p1 \ p2 \ \text{welt}) = \text{welt}$

lemma $\text{zahlenwps-sym: zahlenwps } p1 \ p2 = \text{zahlenwps } p2 \ p1$

definition $\text{abmachungs-betroffene} :: ('person::enum, 'etwas::zero) \text{ abmachung} \Rightarrow 'person \text{ list}$
where

```

    abmachungs-betroffene a  $\equiv [p. p \leftarrow Enum.enum, a \ p \neq 0]$ 

```

lemma $\text{abmachungs-betroffene-is-dom: set } (\text{abmachungs-betroffene } a) = \text{abmachung-dom } a$

lemma $\langle \text{abmachungs-betroffene } (\text{aenderung-map } [\text{Gewinnt Bob } (3::int), \text{Verliert Alice } 3])$
 $= [Alice, Bob] \rangle$

definition $\text{enthaelt-konsens} :: (person, int) \text{ abmachung} \Rightarrow \text{zahlenwelt} \Rightarrow bool$

where

```

    enthaelt-konsens abmachung welt  $\equiv \forall \text{ betroffene-person} \in \text{set } (\text{abmachungs-betroffene abmachung}).$ 

```

```

    abmachung  $\in \text{set } (\text{konsens welt betroffene-person})$ 

```

definition $\text{hat-konsens} :: \text{zahlenwelt} \text{ handlung} \Rightarrow bool$

where

```

    hat-konsens h  $\equiv$ 

```

```

    let abmachung = aenderung-map (delta-num-fun (map-handlung besitz h))

```

in enthaelt-konsens abmachung (vorher h)

lemma *enthaelt-konsens (TODOSWAP delta) (zahlenwps p1 p2 welt) = enthaelt-konsens delta welt*

lemma *hat-konsens (map-handlung (zahlenwps p1 p2) h) = hat-konsens h*

Eine Handlung die keine Änderung bewirkt hat keine Betroffenen und damit immer Konsens.

lemma *hat-konsens (handeln p welt (Handlungsabsicht (λp w. Some w)))*

lemma *hat-konsens (handeln Alice initialwelt*

(Handlungsabsicht (λp w. Some (w ↦ besitz := [[(besitz w)(Alice += 3)](Bob -= 3)])))

lemma *¬ hat-konsens (handeln Alice initialwelt*

(Handlungsabsicht (λp w. Some (w ↦ besitz := [[(besitz w)(Alice += 4)](Bob -= 4)])))

term *Aenderung.abmachung-ausfuehren*

definition *abmachung-ausfuehren*

:: (person, int) abmachung ⇒ zahlenwelt ⇒ zahlenwelt

where

abmachung-ausfuehren abmachung welt ≡ welt ↦ besitz := Aenderung.abmachung-ausfuehren abmachung (besitz welt) ↦

lemma *⟨abmachung-ausfuehren (aenderung-map [Gewinnt Alice 3]) initialwelt*

= initialwelt ↦ besitz := [(besitz initialwelt)(Alice += 3)]⟩

value *⟨remove1 3 [1::int,3,5,2,3]⟩*

value *⟨remove1 9 [1::int,3,5,2,3]⟩*

definition *konsens-entfernen*

:: ('person::enum, 'etwas::zero) abmachung ⇒ ('person ⇒ ('person, 'etwas) abmachung list)

⇒ ('person ⇒ ('person, 'etwas) abmachung list)

where

konsens-entfernen abmachung kons =

fold (λp k. k(p := remove1 abmachung (k p))) (abmachungs-betroffene abmachung) kons

lemma *⟨konsens-entfernen (aenderung-map [Gewinnt Alice 3, Verliert Bob 3]) (konsens initialwelt)*

= (λ-. [])

Alice := [aenderung-map [Gewinnt Alice 3]],

Bob := []⟩

definition *abmachung-einloesen* :: (person, int) *abmachung* \Rightarrow zahlenwelt \Rightarrow zahlenwelt option **where**
abmachung-einloesen delta welt \equiv
 if *enthaelt-konsens* delta welt
 then Some ((*abmachung-ausfuehren* delta welt)() konsens := *konsens-entfernen* delta (konsens welt))
 else None

lemma \langle *abmachung-einloesen* (*aenderung-map* [Gewinnt Alice 3, Verliert Bob 3]) initialwelt
 = Some
 ()
 besitz = \clubsuit [Alice := 8, Bob := 7, Carol := -3],
 konsens = (λ -. [])(
 Alice := [*aenderung-map* [Gewinnt Alice 3]],
 Bob := []),
 staatsbesitz = 9000,
 umwelt = 600
) \rangle

lemma \langle *abmachung-einloesen* (*aenderung-map* [Gewinnt Alice 3]) initialwelt
 = Some
 ()
 besitz = \clubsuit [Alice := 8, Bob := 10, Carol := -3],
 konsens = (λ -. [])(
 Alice := [*aenderung-map* [Gewinnt Alice 3, Verliert Bob 3]],
 Bob := [*aenderung-map* [Gewinnt Alice 3, Verliert Bob 3]]),
 staatsbesitz = 9000,
 umwelt = 600
) \rangle

lemma \langle *abmachung-einloesen* (*aenderung-map* [Verliert Bob 3]) initialwelt = None \rangle

lemma \neg wohlgeformte-handlungsabsicht zahlenwps initialwelt
 (Handlungsabsicht (λ p w. *abmachung-einloesen* (*aenderung-map* [Gewinnt Alice 3]) w))

definition *existierende-abmachung-einloesen* :: person \Rightarrow zahlenwelt \Rightarrow zahlenwelt option **where**
existierende-abmachung-einloesen p welt \equiv
 case (konsens welt) p
 of [] \Rightarrow None
 | d#- \Rightarrow *abmachung-einloesen* d welt

lemma wohlgeformte-handlungsabsicht zahlenwps initialwelt
 (Handlungsabsicht *existierende-abmachung-einloesen*)

lemma map-option (zahlenwps p1 p2) (*abmachung-einloesen* a welt) =

abmachung-einloesen (swap p1 p2 a) (zahlenwps p1 p2 welt)

lemma *map-option (zahlenwps p1 p2) (existierende-abmachung-einloesen p1 welt)*
= existierende-abmachung-einloesen p2 (zahlenwps p1 p2 welt)

lemma *wohlgeformte-handlungsabsicht zahlenwps welt*
(Handlungsabsicht existierende-abmachung-einloesen)

Es ist nur möglich, wenn alle Betroffenen auch zustimmen. Es is beispielsweise nicht möglich, dass *Alice* eine Handlung ausführt, die *Carol* betrifft, ohne deren Zustimmung.

lemma \neg *ausfuehrbar Alice*
 \langle
besitz = \clubsuit [*Alice* := 5, *Bob* := 10, *Carol* := -3],
konsens = $(\lambda\cdot. \langle \rangle)$
Alice := [aenderung-map [Verliert *Carol* 3]]
 \rangle ,
staatsbesitz = 9000,
umwelt = 600
 \rangle
(Handlungsabsicht existierende-abmachung-einloesen)

Nur wenn *Carol* zustimmt wird die Handlung möglich.

lemma *ausfuehrbar Alice*
 \langle
besitz = \clubsuit [*Alice* := 5, *Bob* := 10, *Carol* := -3],
konsens = $(\lambda\cdot. \langle \rangle)$
Alice := [aenderung-map [Verliert *Carol* 3]],
Carol := [aenderung-map [Verliert *Carol* 3]]
 \rangle ,
staatsbesitz = 9000,
umwelt = 600
 \rangle
(Handlungsabsicht existierende-abmachung-einloesen)

Da *Alice* nicht betroffen is, bleibt [Verliert *Carol* (3::'a)] bei *Alice* übrig.

lemma *nachher-handeln Alice*
 \langle
besitz = \clubsuit [*Alice* := 5, *Bob* := 10, *Carol* := -3],
konsens = $(\lambda\cdot. \langle \rangle)$
Alice := [aenderung-map [Verliert *Carol* 3]],
Carol := [aenderung-map [Verliert *Carol* 3]]
 \rangle ,
staatsbesitz = 9000,
umwelt = 600
 \rangle

```

(Handlungsabsicht existierende-abmachung-einloesen)
= ()
  besitz = ♣[Alice := 5, Bob := 10, Carol := -6],
  konsens = (λ-. ()) (
    Alice := [aenderung-map [Verliert Carol 3]],
    Carol := []
  ),
  staatsbesitz = 9000,
  umwelt = 600
)

```

Ressourcen können nicht aus dem Nichts erschaffen werden.

```

fun abbauen :: ⟨nat ⇒ person ⇒ zahlenwelt ⇒ zahlenwelt option⟩ where
  ⟨abbauen i p welt = Some (welt() besitz := [(besitz welt)(p += int i)], umwelt := (umwelt welt) - int i ())⟩

```

```

lemma ⟨wohlgeformte-handlungsabsicht zahlenwps welt (Handlungsabsicht (abbauen n))⟩

```

```

lemma ⟨wohlgeformte-handlungsabsicht zahlenwps initialwelt (Handlungsabsicht (abbauen n))⟩

```

```

fun reset :: ⟨person ⇒ zahlenwelt ⇒ zahlenwelt option⟩ where
  ⟨reset ich welt = Some (welt() besitz := λ -. 0())⟩

```

```

lemma ⟨wohlgeformte-handlungsabsicht zahlenwps welt (Handlungsabsicht reset)⟩

```

```

fun alles-kaputt-machen :: ⟨person ⇒ zahlenwelt ⇒ zahlenwelt option⟩ where
  ⟨alles-kaputt-machen ich welt = Some (welt() besitz := λ -. Min ((besitz welt) ‘UNIV) - 1 ())⟩

```

```

lemma alles-kaputt-machen-code[code]:
  ⟨alles-kaputt-machen ich welt =
    Some (welt() besitz := (λ-. min-list (map (besitz welt) enum-class.enum) - 1)())⟩

```

```

fun unmoeglich :: ⟨person ⇒ zahlenwelt ⇒ zahlenwelt option⟩ where
  ⟨unmoeglich - - = None⟩

```

fun *individueller-fortschritt* :: $\langle \text{person} \Rightarrow \text{zahlenwelt} \text{ handlung} \Rightarrow \text{bool} \rangle$ **where**
 $\langle \text{individueller-fortschritt } p \text{ (Handlung vor nach)} \longleftrightarrow (\text{meins } p \text{ vor}) \leq (\text{meins } p \text{ nach}) \rangle$

definition *maxime-altruistischer-fortschritt* :: $\langle (\text{person}, \text{zahlenwelt}) \text{ maxime} \rangle$ **where**
 $\langle \text{maxime-altruistischer-fortschritt} \equiv$
 $\text{Maxime } (\lambda h. \forall pX. \text{individueller-fortschritt } pX \text{ } h) \rangle$

value_[simp] $\langle \text{erzeuge-beispiel}$
 $\text{zahlenwps initialwelt}$
 $[\text{Handlungsabsicht (abbauen 5)},$
 $\text{Handlungsabsicht existierende-abmachung-einloesen},$
 $\text{Handlungsabsicht reset},$
 $\text{Handlungsabsicht alles-kaputt-machen},$
 $\text{Handlungsabsicht unmoeglich}]$
 $\text{maxime-altruistischer-fortschritt} \rangle$

definition *maxime-hatte-konsens* :: $(\text{person}, \text{zahlenwelt}) \text{ maxime}$ **where**
 $\langle \text{maxime-hatte-konsens} \equiv \text{Maxime } (\lambda h. \text{hat-konsens } h) \rangle$

lemma $\langle \forall h \in \text{set } (\text{alle-moeöglichen-handlungen } \text{initialwelt } [\text{Handlungsabsicht}$
 $\text{existierende-abmachung-einloesen}]) .$
 $\text{wohlgeformte-maxime-auf}$
 $h \text{ zahlenwps}$
 $\text{maxime-hatte-konsens} \rangle$

lemma *wohlgeformte-maxime zahlenwps maxime-hatte-konsens*

lemma $\langle \text{erzeuge-beispiel}$
 $\text{zahlenwps initialwelt}$
 $[\text{Handlungsabsicht existierende-abmachung-einloesen}]$
 $\text{maxime-hatte-konsens}$
 $= \text{Some}$
 $(\text{bsp-welt} = \text{initialwelt},$
 $\text{bsp-erfuellte-maxime} = \text{Some maxime-hatte-konsens},$
 $\text{bsp-erlaubte-handlungen} = [\text{Handlungsabsicht existierende-abmachung-einloesen}],$
 $\text{bsp-verbotene-handlungen} = []) \rangle$

lemma $\langle \text{erzeuge-beispiel}$
 $\text{zahlenwps initialwelt}$
 $[\text{Handlungsabsicht (abbauen 5)},$
 $\text{Handlungsabsicht reset},$

```

    Handlungsabsicht alles-kaputt-machen,
    Handlungsabsicht unmöglich]
maxime-altruistischer-fortschritt
= Some
  (|bsp-welt = initialwelt,
   bsp-erfuellte-maxime = Some maxime-altruistischer-fortschritt,
   bsp-erlaubte-handlungen = [Handlungsabsicht (abbauen 5), Handlungsabsicht unmöglich],
   bsp-verbotene-handlungen = [Handlungsabsicht reset, Handlungsabsicht alles-kaputt-machen])|)

```

```

lemma <erzeuge-beispiel
zahlenwps initialwelt
[Handlungsabsicht (abbauen 5),
 Handlungsabsicht existierende-abmachung-einloesen,
 Handlungsabsicht reset,
 Handlungsabsicht alles-kaputt-machen,
 Handlungsabsicht unmöglich]
(MaximeDisj maxime-altruistischer-fortschritt maxime-hatte-konsens)
= Some
  (|bsp-welt = initialwelt,
   bsp-erfuellte-maxime = Some (MaximeDisj maxime-altruistischer-fortschritt maxime-hatte-konsens),
   bsp-erlaubte-handlungen = [Handlungsabsicht (abbauen 5), Handlungsabsicht
existierende-abmachung-einloesen, Handlungsabsicht unmöglich],
   bsp-verbotene-handlungen = [Handlungsabsicht reset, Handlungsabsicht alles-kaputt-machen])|)

```

14 Gesetz

Definiert einen Datentyp um Gesetzestext zu modellieren.

```
datatype 'a tatbestand = Tatbestand <'a>
```

```
datatype 'a rechtsfolge = Rechtsfolge <'a>
```

```
datatype ('a, 'b) rechtsnorm = Rechtsnorm <'a tatbestand> <'b rechtsfolge>
```

```
datatype 'p prg = Paragraph <'p> (§)
```

```
datatype ('p, 'a, 'b) gesetz = Gesetz <('p prg × ('a, 'b) rechtsnorm) set>
```

Beispiel, von <https://de.wikipedia.org/wiki/Rechtsfolge>:

```

value <Gesetz {
  (§ "823 BGB",
   Rechtsnorm
    (Tatbestand "Wer vorsatzlich oder fahrlaessig das Leben, den Koerper, die Gesundheit, (...),
               das Eigentum oder (...) eines anderen widerrechtlich verletzt,"))
  (Rechtsfolge "ist dem anderen zum Ersatz des daraus entstehenden Schadens verpflichtet.")
  },
  (§ "985 BGB",

```

```

Rechtsnorm
  (Tatbestand "Der Eigentüemer einer Sache kann von dem Besitzer")
  (Rechtsfolge "die Herausgabe der Sache verlangen")
),
 (§ "303 StGB",
  Rechtsnorm
    (Tatbestand "Wer rechtswidrig eine fremde Sache beschadigt oder zerstört,")
    (Rechtsfolge "wird mit Freiheitsstrafe bis zu zwei Jahren oder mit Geldstrafe bestraft."))
)
}⟩

```

```

fun neuer-paragraph :: ⟨(nat, 'a, 'b) gesetz ⇒ nat prg⟩ where
  ⟨neuer-paragraph (Gesetz G) = § ((max-paragraph (fst ‘ G)) + 1)⟩

```

Fügt eine Rechtsnorm als neuen Paragraphen hinzu:

```

fun hinzufuegen :: ⟨('a,'b) rechtsnorm ⇒ (nat,'a,'b) gesetz ⇒ (nat,'a,'b) gesetz⟩ where
  ⟨hinzufuegen rn (Gesetz G) =
    (if rn ∈ (snd ‘ G) then Gesetz G else Gesetz (insert (neuer-paragraph (Gesetz G), rn) G))⟩

```

Modelliert ob eine Handlung ausgeführt werden muss, darf, kann, nicht muss:

```

datatype sollensanordnung = Gebot | Verbot | Erlaubnis | Freistellung

```

Beispiel:

```

lemma ⟨hinzufuegen
  (Rechtsnorm (Tatbestand "tb2") (Rechtsfolge Verbot))
  (Gesetz { (§ 1, (Rechtsnorm (Tatbestand "tb1") (Rechtsfolge Erlaubnis))) }) =
  Gesetz
  { (§ 2, Rechtsnorm (Tatbestand "tb2") (Rechtsfolge Verbot)),
    (§ 1, Rechtsnorm (Tatbestand "tb1") (Rechtsfolge Erlaubnis)) }⟩

```

15 Experimental: Moralisch Gesetzs Ableiten

15.1 Allgemeines Gesetz Ableiten

Wir wollen implementieren:

„Handle nur nach derjenigen *Maxime*, durch die du zugleich wollen kannst, dass sie ein **allgemeines Gesetz** werde.“

Für eine gebene Welt haben wir schon eine Handlung nach einer *Maxime* untersucht: *moralisch*

Das Ergebnis sagt uns ob diese Handlung gut oder schlecht ist. Basierend darauf müssen wir nun ein allgemeines Gesetz ableiten.

Ich habe keine Ahnung wie das genau funktionieren soll, deswegen schreibe ich einfach nur in einer Typsignatur auf, was zu tun ist:

Gegeben:

- *'world handlung*: Die Handlung
- *sollensanordnung*: Das Ergebnis der moralischen Bewertung, ob die Handlung gut/schlecht.

Gesucht:

- (*'a, 'b*) *rechtsnorm*: ein allgemeines Gesetz

type-synonym (*'world, 'a, 'b*) *allgemeines-gesetz-ableiten* =
 $\langle 'world\ handlung \Rightarrow sollensanordnung \Rightarrow ('a, 'b)\ rechtsnorm \rangle$

Soviel vorweg: Nur aus einer von außen betrachteten Handlung und einer Entscheidung ob diese Handlung ausgeführt werden soll wird es schwer ein allgemeines Gesetz abzuleiten.

15.2 Implementierung Moralisch ein Allgemeines Gesetz Ableiten

Und nun werfen wir alles zusammen:

„Handle nur nach derjenigen *Maxime*, durch die du zugleich wollen kannst, dass sie ein allgemeines Gesetz werde.“

Eingabe:

- *'person*: handelnde Person
- *'world*: Die Welt in ihrem aktuellen Zustand
- (*'person, 'world*) *handlungsabsicht*: Eine mögliche Handlung, über die wir entscheiden wollen ob wir sie ausführen sollten.
- (*'person, 'world*) *maxime*: Persönliche Ethik.
- (*'world, 'a, 'b*) *allgemeines-gesetz-ableiten*: wenn man keinen Plan hat wie man sowas implementiert, einfach als Eingabe annehmen.
- (*nat, 'a, 'b*) *gesetz*: Initiales allgemeines Gesetz (normalerweise am Anfang leer).

Ausgabe: *sollensanordnung*: Sollen wir die Handlung ausführen? (*nat, 'a, 'b*) *gesetz*: Soll das allgemeine Gesetz entsprechend angepasst werden?

definition *moarlich-gesetz-ableiten* ::

```

⟨ 'person ⇒
  'world ⇒
    ('person, 'world) maxime ⇒
    ('person, 'world) handlungsabsicht ⇒
    ('world, 'a, 'b) allgemeines-gesetz-ableiten ⇒
    (nat, 'a, 'b) gesetz
  ⇒ (sollensanordnung × (nat, 'a, 'b) gesetz) ⟩

```

where

```

⟨moarlisch-gesetz-ableiten ich welt maxime handlungsabsicht gesetz-ableiten gesetz ≡
  let soll-handeln = if moralisch welt maxime handlungsabsicht
    then
      Erlaubnis
    else
      Verbot in
  (
    soll-handeln,
    hinzufuegen (gesetz-ableiten (handeln ich welt handlungsabsicht) soll-handeln) gesetz
  )⟩

```

Das ganze *moarlisch-gesetz-ableiten* dient mehr dem Debugging, ...

16 Gesetze

Wir implementieren Strategien um (*'world*, *'a*, *'b*) *allgemeines-gesetz-ableiten* zu implementieren.

16.1 Case Law Absolut

Gesetz beschreibt: wenn (vorher, nachher) dann Erlaubt/Verboten, wobei vorher/nachher die Welt beschreiben. Paragraphen sind einfache natürliche Zahlen.

type-synonym *'world case-law* = ⟨(*nat*, (*'world* × *'world*), *sollensanordnung*) gesetz⟩

Überträgt einen Tatbestand wörtlich ins Gesetz. Nicht sehr allgemein.

definition *case-law-ableiten-absolut*

:: ⟨(*'world*, (*'world* × *'world*), *sollensanordnung*) allgemeines-gesetz-ableiten⟩

where

⟨*case-law-ableiten-absolut handlung sollensanordnung* =
Rechtsnorm
 (*Tatbestand* (*vorher handlung*, *nachher handlung*))
 (*Rechtsfolge sollensanordnung*)⟩

definition *printable-case-law-ableiten-absolut*

:: ⟨(*'world* ⇒ *'printable-world*) ⇒
 (*'world*, (*'printable-world* × *'printable-world*), *sollensanordnung*) allgemeines-gesetz-ableiten⟩

where

⟨*printable-case-law-ableiten-absolut print-world h* ≡
case-law-ableiten-absolut (*map-handlung print-world h*)⟩

16.2 Case Law Relativ

Case Law etwas besser, wir zeigen nur die Änderungen der Welt.

fun *case-law-ableiten-relativ*

:: ⟨(*'world handlung* ⇒ ((*'person*, *'etwas*) *aenderung*) *list*)
 ⇒ (*'world*, ((*'person*, *'etwas*) *aenderung*) *list*, *sollensanordnung*)
allgemeines-gesetz-ableiten⟩

where

⟨*case-law-ableiten-relativ delta handlung erlaubt* =
Rechtsnorm (Tatbestand (delta handlung)) (Rechtsfolge erlaubt)⟩

17 Simulation

Gegeben eine handelnde Person und eine Maxime, wir wollen simulieren was für ein allgemeines Gesetz abgeleitet werden könnte.

datatype (*'person, 'world, 'a, 'b*) *simulation-constants* = *SimConsts*
 ⟨*'person*⟩ — handelnde Person
 ⟨(*'person, 'world*) *maxime*⟩
 ⟨(*'world, 'a, 'b*) *allgemeines-gesetz-ableiten*⟩

...

... Die Funktion *simulateOne* nimmt eine Konfiguration (*'person, 'world, 'a, 'b*) *simulation-constants*, eine Anzahl an Iterationen die durchgeführt werden sollen, eine Handlung, eine Initialwelt, ein Initialgesetz, und gibt das daraus resultierende Gesetz nach so vielen Iterationen zurück.

Beispiel: Wir nehmen die mir-ist-alles-egal Maxime. Wir leiten ein allgemeines Gesetz ab indem wir einfach nur die Handlung wörtlich ins Gesetz übernehmen. Wir machen *10::'a* Iterationen. Die Welt ist nur eine Zahl und die initiale Welt sei *32::'a*. Die Handlung ist es diese Zahl um Eins zu erhöhen, Das Ergebnis der Simulation ist dann, dass wir einfach von *32::'a* bis *42::'a* zählen.

lemma *simulateOne*
 (*SimConsts* () (*Maxime* (λ-. True)) (λ*h s*. *Rechtsnorm (Tatbestand h) (Rechtsfolge "count")*))
 10 (*Handlungsabsicht* (λ*p n*. *Some (Suc n)*))
 32
 (*Gesetz* {}) =
Gesetz
 {(§ 10, *Rechtsnorm (Tatbestand (Handlung 41 42)) (Rechtsfolge "count")*),
 (§ 9, *Rechtsnorm (Tatbestand (Handlung 40 41)) (Rechtsfolge "count")*),
 (§ 8, *Rechtsnorm (Tatbestand (Handlung 39 40)) (Rechtsfolge "count")*),
 (§ 7, *Rechtsnorm (Tatbestand (Handlung 38 39)) (Rechtsfolge "count")*),
 (§ 6, *Rechtsnorm (Tatbestand (Handlung 37 38)) (Rechtsfolge "count")*),
 (§ 5, *Rechtsnorm (Tatbestand (Handlung 36 37)) (Rechtsfolge "count")*),
 (§ 4, *Rechtsnorm (Tatbestand (Handlung 35 36)) (Rechtsfolge "count")*),
 (§ 3, *Rechtsnorm (Tatbestand (Handlung 34 35)) (Rechtsfolge "count")*),
 (§ 2, *Rechtsnorm (Tatbestand (Handlung 33 34)) (Rechtsfolge "count")*),
 (§ 1, *Rechtsnorm (Tatbestand (Handlung 32 33)) (Rechtsfolge "count")*)}

Eine Iteration der Simulation liefert genau einen Paragraphen im Gesetz:

lemma *tb rf*.
simulateOne
 (*SimConsts person maxime gesetz-ableiten*)
 1 *handlungsabsicht*
initialwelt

$(Gesetz \{\})$
 $= Gesetz \{(\S 1, Rechtsnorm (Tatbestand tb) (Rechtsfolge rf))\}$

18 Beispiel: BeispielZahlenwelt aber mit Gesetz (Experimental)

18.1 Setup

Wir nehmen an unsere handelnde Person ist *Alice*.

definition $\langle beispiel-case-law-absolut maxime handlungsabsicht \equiv$
 $simulateOne$
 $(SimConsts$
 $Alice$
 $maxime$
 $(printable-case-law-ableiten-absolut show-zahlenwelt))$
 $5 handlungsabsicht initialwelt (Gesetz \{\}) \rangle$

definition $\langle beispiel-case-law-relativ maxime handlungsabsicht \equiv$
 $simulateOne$
 $(SimConsts$
 $Alice$
 $maxime$
 $(case-law-ableiten-relativ delta-zahlenwelt))$
 $10 handlungsabsicht initialwelt (Gesetz \{\}) \rangle$

18.2 Beispiele

Alice kann beliebig oft 5 Wohlstand für sich selbst erschaffen. Das entstehende Gesetz ist nicht sehr gut, da es einfach jedes Mal einen Snapshot der Welt aufschreibt und nicht sehr generisch ist.

lemma $\langle beispiel-case-law-absolut maxime-zahlenfortschritt (Handlungsabsicht (erschaffen 5))$
 $=$
 $Gesetz$
 $\{(\S 5,$
 $Rechtsnorm$
 $(Tatbestand ([(Alice, 25), (Bob, 10), (Carol, - 3)], [(Alice, 30), (Bob, 10), (Carol, - 3)]))$
 $(Rechtsfolge Erlaubnis)),$
 $(\S 4,$
 $Rechtsnorm$
 $(Tatbestand ([(Alice, 20), (Bob, 10), (Carol, - 3)], [(Alice, 25), (Bob, 10), (Carol, - 3)]))$
 $(Rechtsfolge Erlaubnis)),$
 $(\S 3,$
 $Rechtsnorm$
 $(Tatbestand ([(Alice, 15), (Bob, 10), (Carol, - 3)], [(Alice, 20), (Bob, 10), (Carol, - 3)]))$
 $(Rechtsfolge Erlaubnis)),$
 $(\S 2,$
 $Rechtsnorm$
 $(Tatbestand ([(Alice, 10), (Bob, 10), (Carol, - 3)], [(Alice, 15), (Bob, 10), (Carol, - 3)]))$
 $(Rechtsfolge Erlaubnis)),$
 $(\S 1,$

Rechtsnorm
 (Tatbestand $((Alice, 5), (Bob, 10), (Carol, -3)), ((Alice, 10), (Bob, 10), (Carol, -3)))$
 (Rechtsfolge Erlaubnis))
 \rangle

Die gleiche Handlung, wir schreiben aber nur die Änderung der Welt ins Gesetz:

lemma \langle *beispiel-case-law-relativ maxime-zahlenfortschritt* (Handlungsabsicht (erschaffen 5)) =
 Gesetz
 { (§ 1, Rechtsnorm (Tatbestand [Gewinnt Alice 5]) (Rechtsfolge Erlaubnis)) } \rangle

lemma \langle *beispiel-case-law-relativ*
 (Maxime $(\lambda(ich::person) h. (\forall pX. \text{individueller-fortschritt } pX h))$) (Handlungsabsicht (erschaffen 5)) =
 Gesetz { (§ 1, Rechtsnorm (Tatbestand [Gewinnt Alice 5]) (Rechtsfolge Erlaubnis)) } \rangle

Nun ist es Alice verboten Wohlstand für sich selbst zu erzeugen.

lemma \langle *beispiel-case-law-relativ*
 (Maxime $(\lambda ich. \text{individueller-strikter-fortschritt } ich)$)
 (Handlungsabsicht (erschaffen 5)) =
 Gesetz { (§ 1, Rechtsnorm (Tatbestand [Gewinnt Alice 5]) (Rechtsfolge Verbot)) } \rangle

lemma \langle *beispiel-case-law-relativ*
 (Maxime $(\lambda ich. \text{globaler-strikter-fortschritt})$)
 (Handlungsabsicht (erschaffen 5)) =
 Gesetz { (§ 1, Rechtsnorm (Tatbestand [Gewinnt Alice 5]) (Rechtsfolge Erlaubnis)) } \rangle

lemma \langle *beispiel-case-law-relativ*
 (Maxime $(\lambda ich. \text{globaler-strikter-fortschritt})$)
 (Handlungsabsicht (erschaffen 0)) =
 Gesetz { (§ 1, Rechtsnorm (Tatbestand []) (Rechtsfolge Verbot)) } \rangle

lemma \langle *beispiel-case-law-relativ*
maxime-zahlenfortschritt
 (Handlungsabsicht (erschaffen 0)) =
 Gesetz { (§ 1, Rechtsnorm (Tatbestand []) (Rechtsfolge Erlaubnis)) } \rangle

lemma \langle *beispiel-case-law-relativ*
 (Maxime $(\lambda ich. \text{globaler-fortschritt})$)
 (Handlungsabsicht (erschaffen 0))

=
Gesetz { (§ 1, *Rechtsnorm* (*Tatbestand* []) (*Rechtsfolge Erlaubnis*)) } ›

lemma ‹*beispiel-case-law-relativ*
 (*Maxime* (λ ich. *globaler-fortschritt*))
 (*Handlungsabsicht* (*stehlen-nichtwf* 5 *Bob*))

=
Gesetz
 { (§ 1, *Rechtsnorm* (*Tatbestand* [*Gewinnt Alice* 5, *Verliert Bob* 5]) (*Rechtsfolge Erlaubnis*)) } ›

Stehlen ist verboten:

lemma ‹*beispiel-case-law-relativ maxime-zahlenfortschritt* (*Handlungsabsicht* (*stehlen-nichtwf* 5 *Bob*)) =

Gesetz
 { (§ 1, *Rechtsnorm* (*Tatbestand* [*Gewinnt Alice* 5, *Verliert Bob* 5]) (*Rechtsfolge Verbot*)) } ›

Auch wenn *Alice* von sich selbst stehlen möchte ist dies verboten, obwohl hier keiner etwas verliert:

lemma ‹*beispiel-case-law-relativ maxime-zahlenfortschritt* (*Handlungsabsicht* (*stehlen-nichtwf* 5 *Alice*)) =

Gesetz { (§ 1, *Rechtsnorm* (*Tatbestand* []) (*Rechtsfolge Verbot*)) } ›

Der Grund ist, dass *Alice* die abstrakte Handlung "Alice wird bestohlen" gar nicht gut fände, wenn sie jemand anderes ausführt:

lemma ‹*debug-maxime show-zahlenwelt initialwelt*
maxime-zahlenfortschritt (*Handlungsabsicht* (*stehlen-nichtwf* 5 *Alice*)) =

{ *VerletzteMaxime* (*Opfer Alice*) (*Taeter Bob*)
 (*Handlung* [(*Alice*, 5), (*Bob*, 10), (*Carol*, - 3)] [(*Bob*, 15), (*Carol*, - 3)]),
VerletzteMaxime (*Opfer Alice*) (*Taeter Carol*)
 (*Handlung* [(*Alice*, 5), (*Bob*, 10), (*Carol*, - 3)] [(*Bob*, 10), (*Carol*, 2)]),
VerletzteMaxime (*Opfer Alice*) (*Taeter Eve*)
 (*Handlung* [(*Alice*, 5), (*Bob*, 10), (*Carol*, - 3)] [(*Bob*, 10), (*Carol*, - 3), (*Eve*, 5)])
 } ›

Leider ist das hier abgeleitete Gesetz sehr fragwürdig: *Rechtsnorm* (*Tatbestand* []) (*Rechtsfolge Verbot*)

Es besagt, dass Nichtstun verboten ist.

Indem wir die beiden Handlungen Nichtstun und Selbstbestehlen betrachten, können wir sogar ein widersprüchliches Gesetz ableiten:

lemma ‹*simulateOne*
 (*Sim Consts*
Alice
maxime-zahlenfortschritt
 (*case-law-ableiten-relativ delta-zahlenwelt*))

$$\begin{aligned}
& 20 \text{ (Handlungsabsicht (stehlen-nichtwf 5 Alice)) initialwelt} \\
& \text{(beispiel-case-law-relativ maxime-zahlenfortschritt (Handlungsabsicht (erschaffen 0)))} \\
& = \\
& \text{Gesetz} \\
& \{(\S 2, \text{Rechtsnorm (Tatbestand [] (Rechtsfolge Verbot)),} \\
& (\S 1, \text{Rechtsnorm (Tatbestand []) (Rechtsfolge Erlaubnis))\} \rangle
\end{aligned}$$

Meine persönliche Conclusion: Wir müssen irgendwie die Absicht mit ins Gesetz schreiben.

Es ist *Alice* verboten, etwas zu verschenken:

$$\begin{aligned}
& \text{lemma} \langle \text{beispiel-case-law-relativ maxime-zahlenfortschritt (Handlungsabsicht (schenken 5 Bob))} \\
& = \\
& \text{Gesetz} \\
& \{(\S 1, \\
& \quad \text{Rechtsnorm (Tatbestand [Verliert Alice 5, Gewinnt Bob 5]) (Rechtsfolge Verbot))\} \rangle
\end{aligned}$$

Der Grund ist, dass *Alice* dabei etwas verliert und die *maxime-zahlenfortschritt* dies nicht Erlaubt. Es fehlt eine Möglichkeit zu modellieren, dass *Alice* damit einverstanden ist, etwas abzugeben. Doch wir haben bereits in *stehlen-nichtwf i = schenken (− i)* gesehen, dass *stehlen* und *schenken* nicht unterscheidbar sind.

Folgende ungültige Maxime würde es erlauben, dass *Alice* Leute bestehlen darf:

$$\begin{aligned}
& \text{lemma} \langle \text{beispiel-case-law-relativ} \\
& \quad \text{(Maxime (\lambda ich. individueller-fortschritt Alice))} \\
& \quad \text{(Handlungsabsicht (stehlen-nichtwf 5 Bob))} \\
& = \\
& \text{Gesetz} \\
& \{(\S 1, \text{Rechtsnorm (Tatbestand [Gewinnt Alice 5, Verliert Bob 5]) (Rechtsfolge Erlaubnis))\} \rangle
\end{aligned}$$

19 Einkommensteuergesetzgebung

Basierend auf einer stark vereinfachten Version des deutschen Steuerrechts. Wenn ich Wikipedia richtig verstanden habe, habe ich sogar aus Versehen einen Teil des österreichischen Steuersystem gebaut mit deutschen Konstanten.

Folgende **locale** nimmt an, dass wir eine Funktion *steuer::nat* \Rightarrow *nat* haben, welche basierend auf dem Einkommen die zu zahlende Steuer berechnet.

Die *steuer* Funktion arbeitet auf natürlichen Zahlen. Wir nehmen an, dass einfach immer auf ganze Geldbeträge gerundet wird. Wie im deutschen System.

Die **locale** enthält einige Definition, gegeben die *steuer* Funktion.

Eine konkrete *steuer* Funktion wird noch nicht gegeben.

$$\begin{aligned}
& \text{locale steuer-defs =} \\
& \quad \text{fixes steuer :: } \langle \text{nat} \Rightarrow \text{nat} \rangle \text{ — Einkommen -> Steuer}
\end{aligned}$$

```

begin
  definition brutto ::  $\langle \text{nat} \Rightarrow \text{nat} \rangle$  where
     $\langle \text{brutto einkommen} \equiv \text{einkommen} \rangle$ 
  definition netto ::  $\langle \text{nat} \Rightarrow \text{nat} \rangle$  where
     $\langle \text{netto einkommen} \equiv \text{einkommen} - (\text{steuer einkommen}) \rangle$ 
  definition steuersatz ::  $\langle \text{nat} \Rightarrow \text{percentage} \rangle$  where
     $\langle \text{steuersatz einkommen} \equiv \text{percentage } ((\text{steuer einkommen}) / \text{einkommen}) \rangle$ 
end

```

Beispiel. Die *steuer* Funktion sagt, man muss 25 Prozent Steuern zahlen:

```

definition beispiel-25prozent-steuer ::  $\langle \text{nat} \Rightarrow \text{nat} \rangle$  where
   $\langle \text{beispiel-25prozent-steuer } e \equiv \text{nat } \lfloor \text{real } e * (\text{percentage } 0.25) \rfloor \rangle$ 

```

```

lemma
   $\langle \text{beispiel-25prozent-steuer } 100 = 25 \rangle$ 
   $\langle \text{steuer-defs.brutto } 100 = 100 \rangle$ 
   $\langle \text{steuer-defs.netto beispiel-25prozent-steuer } 100 = 75 \rangle$ 
   $\langle \text{steuer-defs.steuersatz beispiel-25prozent-steuer } 100 = \text{percentage } 0.25 \rangle$ 

```

Folgende **locale** erweitert die *steuer-defs locale* und stellt einige Anforderungen die eine gültige *steuer* Funktion erfüllen muss.

- Wer mehr Einkommen hat, muss auch mehr Steuern zahlen.
- Leistung muss sich lohnen: Wer mehr Einkommen hat muss auch nach Abzug der Steuer mehr übrig haben.
- Existenzminimum: Es gibt ein Existenzminimum, welches nicht besteuert werden darf.

```

locale steuersystem = steuer-defs +
assumes wer-hat-der-gibt:
   $\langle \text{einkommen-a} \geq \text{einkommen-b} \implies \text{steuer einkommen-a} \geq \text{steuer einkommen-b} \rangle$ 

```

```

and leistung-lohnt-sich:
   $\langle \text{einkommen-a} \geq \text{einkommen-b} \implies \text{netto einkommen-a} \geq \text{netto einkommen-b} \rangle$ 

```

— Ein Existenzminimum wird nicht versteuert. Zahl Deutschland 2022, vermutlich sogar die falsche Zahl.

```

and existenzminimum:
   $\langle \text{einkommen} \leq 9888 \implies \text{steuer einkommen} = 0 \rangle$ 

```

```
begin
```

```
end
```

Eigentlich hätte ich gerne noch eine weitere Anforderung.
<https://de.wikipedia.org/wiki/Steuerprogression> sagt "Steuerprogression bedeutet das Ansteigen des Steuersatzes in Abhängigkeit vom zu versteuernden Einkommen oder Vermögen."

Formal betrachtet würde das bedeuten $einkommen-b \leq einkommen-a \implies (\lambda x. \text{real-of-percentage } (steuer-defs.steuersatz \ einkommen-b \ x)) \leq (\lambda x. \text{real-of-percentage } (steuer-defs.steuersatz \ einkommen-a \ x))$

Leider haben wir bereits jetzt in dem Modell eine Annahme getroffen, die es uns quasi unmöglich macht, ein Steuersystem zu implementieren, welches die Steuerprogression erfüllt. Der Grund ist, dass wir die Steuerfunktion auf ganzen Zahlen definiert haben. Aufgrund von Rundung können wir also immer Fälle haben, indem ein höheres Einkommen einen leicht geringeren Steuersatz hat als ein geringeres Einkommen. Beispielsweise bedeutet das für *beispiel-25prozent-steuer*, dass jemand mit 100 EUR Einkommen genau 25 Prozent Steuer zahlt, jemand mit 103 EUR Einkommen aber nur ca 24,3 Prozent Steuer zahlt.

lemma

```

⟨steuer-defs.steuersatz beispiel-25prozent-steuer 100 = percentage 0.25⟩
⟨steuer-defs.steuersatz beispiel-25prozent-steuer 103 = percentage (25 / 103)⟩
⟨percentage (25 / 103) < percentage 0.25⟩
⟨(103::nat) > 100⟩

```

In der Praxis sollten diese kleinen Rundungsfehler kein Problem darstellen, in diesem theoretischen Modell sorgen sie aber dafür, dass unser Steuersystem (und wir modellieren eine vereinfachte Version des deutschen Steuersystems) keine Steuerprogression erfüllt.

Die folgende Liste, basierend auf [https://de.wikipedia.org/wiki/Einkommensteuer_\(Deutschland\)#Tarif_2022](https://de.wikipedia.org/wiki/Einkommensteuer_(Deutschland)#Tarif_2022), sagt in welchem Bereich welcher Prozentsatz an Steuern zu zahlen ist. Beispielsweise sind die ersten 10347 steuerfrei.

definition *steuerbuckets2022* :: ⟨(nat × percentage) list⟩ **where**

```

⟨steuerbuckets2022 ≡ [
    (10347, percentage 0),
    (14926, percentage 0.14),
    (58596, percentage 0.2397),
    (277825, percentage 0.42)
]⟩

```

Für jedes Einkommen über 277825 gilt der Spitzensteuersatz von 45 Prozent. Wir ignorieren die Progressionsfaktoren in Zone 2 und 3.

Folgende Funktion berechnet die zu zahlende Steuer, basierend auf einer Steuerbucketliste.

fun *bucketsteuerAbs* :: ⟨(nat × percentage) list ⇒ percentage ⇒ nat ⇒ real⟩ **where**

```

⟨bucketsteuerAbs ((bis, prozent)#mehr) spitzensteuer e =
    ((min bis e) * prozent)
    + (bucketsteuerAbs (map (λ(s,p). (s-bis,p)) mehr) spitzensteuer (e - bis))⟩
| ⟨bucketsteuerAbs [] spitzensteuer e = e*spitzensteuer⟩

```

Die Einkommenssteuerberechnung, mit Spitzensteuersatz 45 Prozent und finalem Abrunden.

definition *einkommenssteuer* :: ⟨nat ⇒ nat⟩ **where**

```

⟨einkommenssteuer einkommen ≡
    floor (bucketsteuerAbs steuerbuckets2022 (percentage 0.45) einkommen)⟩

```

Beispiel. Alles unter dem Existenzminimum ist steuerfrei:

lemma $\langle \text{einkommenssteuer } 10 = 0 \rangle$

lemma $\langle \text{einkommenssteuer } 10000 = 0 \rangle$

Für ein Einkommen nur knapp über dem Existenzminimum fällt sehr wenig Steuer an:

lemma $\langle \text{einkommenssteuer } 14000 = \text{floor } ((14000 - 10347) * 0.14) \rangle$

lemma $\langle \text{einkommenssteuer } 14000 = 511 \rangle$

Bei einem Einkommen von 20000 EUR wird ein Teil bereits mit den höheren Steuersatz der 3. Zone besteuert:

lemma $\langle \text{einkommenssteuer } 20000 = 1857 \rangle$

lemma $\langle \text{einkommenssteuer } 20000 =$
 $\text{floor } ((14926 - 10347) * 0.14 + (20000 - 14926) * 0.2397) \rangle$

Höhere Einkommen führen zu einer höheren Steuer:

lemma $\langle \text{einkommenssteuer } 40000 = 6651 \rangle$

lemma $\langle \text{einkommenssteuer } 60000 = 11698 \rangle$

Die *einkommenssteuer* Funktion erfüllt die Anforderungen an *steuersystem*.

interpretation *steuersystem*

where *steuer* = $\langle \text{einkommenssteuer} \rangle$

20 Beispiel: Steuern

Wir nehmen eine einfache Welt an, in der jeder Person ihr Einkommen zugeordnet wird.

Achtung: Im Unterschied zum BeispielZahlenwelt.thy modellieren wir hier nicht den Gesamtbesitz, sondern das Jahreseinkommen. Besitz wird ignoriert.

datatype *steuerwelt* = *Steuerwelt*

(*get-einkommen*: $\langle \text{person} \Rightarrow \text{int} \rangle$) — einkommen jeder Person (im Zweifel 0).

fun *steuerwps* :: $\langle \text{person} \Rightarrow \text{person} \Rightarrow \text{steuerwelt} \Rightarrow \text{steuerwelt} \rangle$ **where**

$\langle \text{steuerwps } p1 \ p2 \ (\text{Steuerwelt } \text{besitz}) = \text{Steuerwelt } (\text{swap } p1 \ p2 \ \text{besitz}) \rangle$

fun *steuerlast* :: $\langle \text{person} \Rightarrow \text{steuerwelt } \text{handlung} \Rightarrow \text{int} \rangle$ **where**

$\langle \text{steuerlast } p \ (\text{Handlung } \text{vor } \text{nach}) = ((\text{get-einkommen } \text{vor}) \ p) - ((\text{get-einkommen } \text{nach}) \ p) \rangle$

fun *brutto* :: $\langle \text{person} \Rightarrow \text{steuerwelt } \text{handlung} \Rightarrow \text{int} \rangle$ **where**

$\langle \text{brutto } p \ (\text{Handlung } \text{vor } \text{nach}) = (\text{get-einkommen } \text{vor}) \ p \rangle$

fun *netto* :: $\langle \text{person} \Rightarrow \text{steuerwelt } \text{handlung} \Rightarrow \text{int} \rangle$ **where**

$\langle \text{netto } p \ (\text{Handlung } \text{vor } \text{nach}) = (\text{get-einkommen } \text{nach}) \ p \rangle$

lemma $\langle \text{steuerlast Alice (Handlung (Steuerwelt } \clubsuit[Alice:=8]) (Steuerwelt \clubsuit[Alice:=5])) = 3} \rangle$
lemma $\langle \text{steuerlast Alice (Handlung (Steuerwelt } \clubsuit[Alice:=8]) (Steuerwelt \clubsuit[Alice:=0])) = 8} \rangle$
lemma $\langle \text{steuerlast Bob (Handlung (Steuerwelt } \clubsuit[Alice:=8]) (Steuerwelt \clubsuit[Alice:=5])) = 0} \rangle$
lemma $\langle \text{steuerlast Alice (Handlung (Steuerwelt } \clubsuit[Alice:=-3]) (Steuerwelt \clubsuit[Alice:=-4])) = 1} \rangle$
lemma $\langle \text{steuerlast Alice (Handlung (Steuerwelt } \clubsuit[Alice:=1]) (Steuerwelt \clubsuit[Alice:=-1])) = 2} \rangle$

fun *mehrverdiener* :: $\langle \text{person} \Rightarrow \text{steuerwelt handlung} \Rightarrow \text{person set} \rangle$ **where**
 $\langle \text{mehrverdiener ich (Handlung vor nach) = } \{p. (\text{get-einkommen vor}) p \geq (\text{get-einkommen vor}) \text{ ich} \} \rangle$

lemma $\langle \text{mehrverdiener Alice}$
 $(\text{Handlung (Steuerwelt } \clubsuit[Alice:=8, Bob:=12, Eve:=7]) (Steuerwelt \clubsuit[Alice:=5]))$
 $= \{Alice, Bob\} \rangle$

lemma *mehrverdiener-betrachtet-nur-ausgangszustand*:
 $\langle \text{mehrverdiener } p (\text{handeln } p' \text{ welt } h) = \text{mehrverdiener } p (\text{Handlung welt welt}) \rangle$

Folgende Maxime versucht Steuergerechtigkeit festzuschreiben:

definition *maxime-steuern* :: $\langle (\text{person, steuerwelt}) \text{ maxime} \rangle$ **where**
 $\langle \text{maxime-steuern} \equiv \text{Maxime}$
 $(\lambda \text{ich handlung.}$
 $(\forall p \in \text{mehrverdiener ich handlung.}$
 $\text{steuerlast ich handlung} \leq \text{steuerlast } p \text{ handlung})$
 $\wedge (\forall p \in \text{mehrverdiener ich handlung.}$
 $\text{netto ich handlung} \leq \text{netto } p \text{ handlung})$
 $\rangle \rangle$

fun *delta-steuerwelt* :: $\langle (\text{steuerwelt, person, int}) \text{ delta} \rangle$ **where**
 $\langle \text{delta-steuerwelt (Handlung vor nach) =}$
 $\text{Aenderung.delta-num-fun (Handlung (get-einkommen vor) (get-einkommen nach))} \rangle$

lemma $\langle \text{wpsm-kommutiert (Maxime}$
 $(\lambda \text{ich handlung.}$
 $(\forall p \in \text{mehrverdiener ich handlung.}$
 $\text{steuerlast ich handlung} \leq \text{steuerlast } p \text{ handlung})) \rangle \text{ steuerwps welt} \rangle$

lemma *wfh-steuerberechnung-jeder-zahlt-int*:
 $\langle \text{ha} = \text{Handlungsabsicht } (\lambda \text{ich } w. \text{Some (Steuerwelt ((}\lambda e. e - \text{steuerberechnung } e) \circ (\text{get-einkommen } w)))$
 $\implies \text{wohlgeformte-handlungsabsicht steuerwps welt ha} \rangle$

Wenn die Steuerfunktion monoton ist, dann kann ich auch einen sehr eingeschränkten kat imp zeigen.

lemma \langle
 $(\bigwedge e1\ e2. e1 \leq e2 \implies \text{steuerberechnung } e1 \leq \text{steuerberechnung } e2) \implies$
 $ha = \text{Handlungsabsicht } (\lambda ich\ w. \text{Some } (\text{Steuerwelt } ((\lambda e. e - \text{steuerberechnung } e) \circ (\text{get-einkommen } w))))$
 \implies
kategorischer-imperativ-auf ha welt
(Maxime
($\lambda ich\ handlung.$
($\forall p \in \text{mehrverdiener } ich\ handlung.$
 *$\text{steuerlast } ich\ handlung \leq \text{steuerlast } p\ handlung))$))
 \rangle*

20.1 Setup für Beispiele

definition $\langle \text{initialwelt} \equiv \text{Steuerwelt } \clubsuit[\text{Alice}:=8, \text{Bob}:=3, \text{Eve}:=5] \rangle$

20.2 Beispiel: Keiner Zahlt Steuern

Die Maxime ist erfüllt, da wir immer nur kleiner-gleich fordern!

lemma $\langle \text{moralisch initialwelt maxime-steuern } (\text{Handlungsabsicht } (\lambda ich\ welt. \text{Some } welt)) \rangle$

20.3 Beispiel: Ich zahle 1 Steuer

Das funktioniert nicht:

definition $\langle \text{ich-zahle-1-steuer } ich\ welt \equiv$
 $\text{Some } (\text{Steuerwelt } \llbracket (\text{get-einkommen } welt)(ich - 1) \rrbracket) \rangle$

lemma $\langle \neg \text{moralisch initialwelt maxime-steuern } (\text{Handlungsabsicht } \text{ich-zahle-1-steuer}) \rangle$

Denn jeder muss Steuer zahlen! Ich finde es super spannend, dass hier faktisch ein Gleichbehandlungsgrundsatz rausfällt, ohne dass wir soewtas jemals explizit gefordert haben.

20.4 Beispiel: Jeder zahle 1 Steuer

Jeder muss steuern zahlen: funktioniert, ist aber doof, denn am Ende sind alle im Minus.

Das *ich* wird garnicht verwendet, da jeder Steuern zahlt.

definition $\langle \text{jeder-zahle-1-steuer } ich\ welt \equiv$
 $\text{Some } (\text{Steuerwelt } ((\lambda e. e - 1) \circ (\text{get-einkommen } welt))) \rangle$

lemma $\langle \text{moralisch initialwelt maxime-steuern } (\text{Handlungsabsicht } \text{jeder-zahle-1-steuer}) \rangle$

20.5 Beispiel: Vereinfachtes Deutsches Steuersystem

Jetzt kommt die Steuern.thy ins Spiel.

definition $\text{jeder-zahlt} :: \langle (\text{nat} \Rightarrow \text{nat}) \Rightarrow 'a \Rightarrow \text{steuerwelt} \Rightarrow \text{steuerwelt} \rangle$ **where**
 $\langle \text{jeder-zahlt } \text{steuerberechnung } ich\ welt \equiv$

Steuerwelt $((\lambda e. e - \text{steuerberechnung } e) \circ \text{nat} \circ (\text{get-einkommen } \text{welt}))$

definition $\langle \text{jeder-zahlt-einkommenssteuer } p \ w \equiv \text{Some } (\text{jeder-zahlt-einkommenssteuer } p \ w) \rangle$

Bei dem geringen Einkommen der *initialwelt* zahlt keiner Steuern.

lemma $\langle \text{moralisch } \text{initialwelt } \text{maxime-steuern } (\text{Handlungsabsicht } \text{jeder-zahlt-einkommenssteuer}) \rangle$

Für höhere Einkommen erhalten wir plausible Werte und niemand rutscht ins negative:

lemma $\langle \text{moralisch}$
 $(\text{Steuerwelt } \clubsuit[\text{Alice}:=10000, \text{Bob}:=14000, \text{Eve}:=20000])$
 maxime-steuern
 $(\text{Handlungsabsicht } \text{jeder-zahlt-einkommenssteuer}) \rangle$

lemma $\langle \text{delta-steuerwelt}$
 $(\text{handeln}$
 $\text{Alice } (\text{Steuerwelt } \clubsuit[\text{Alice}:=10000, \text{Bob}:=14000, \text{Eve}:=20000])$
 $(\text{Handlungsabsicht } \text{jeder-zahlt-einkommenssteuer}))$
 $= [\text{Verliert Bob } 511, \text{Verliert Eve } 1857] \rangle$

21 Vereinfachtes Deutsches Steuersystem vs. die Steuermaxime

Die Anforderungen für ein *steuersystem* und die *maxime-steuern* sind vereinbar.

lemma *steuersystem-imp-maxime*:
 $\langle \text{steuersystem } \text{steuersystem-impl} \implies$
 $(\forall \text{welt. moralisch } \text{welt}$
 maxime-steuern
 $(\text{Handlungsabsicht } (\lambda p \ w. \text{Some } (\text{jeder-zahlt } \text{steuersystem-impl } p \ w)))) \rangle$

Mit genug zusätzlichen Annahmen gilt auch die Rückrichtung:

lemma *maxime-imp-steuersystem*:
 $\langle (\forall \text{einkommen. } \text{steuersystem-impl } \text{einkommen} \leq \text{einkommen}) \implies$
 $(\forall \text{einkommen. } \text{einkommen} \leq 9888 \implies \text{steuersystem-impl } \text{einkommen} = 0) \implies$
 $\forall \text{welt. moralisch } \text{welt } \text{maxime-steuern } (\text{Handlungsabsicht } (\lambda p \ w. \text{Some } (\text{jeder-zahlt } \text{steuersystem-impl } p \ w)))$
 $\implies \text{steuersystem } \text{steuersystem-impl} \rangle$

Dass die eine Richtung gilt (Maxime impliziert *steuersystem*), die andere Richtung (*steuersystem* impliziert Maxime) jedoch nicht ohne weitere Annahmen, stimmt auch mit Russels Beobachtung überein: "Kants Maxime [das allgemeine Konzept, nicht meine Implementierung] scheint tatsächlich ein notwendiges, jedoch nicht *ausreichendes* Kriterium der Tugens zu geben" [1]. Insbesondere Russels Folgesatz freut mich, da er mir bestätigt, dass unsere extensionale Betrachtung von Handlungen vielversprechend ist: "Um ein ausreichendes Kriterium zu gewinnen, müßten wir Kants rein formalen Standpunkt aufgeben und die Wirkung der Handlungen in Betracht ziehen" [1].

Für jedes *steuersystem-impl::nat* \Rightarrow *nat*, mit zwei weiteren Annahmen, gilt das *steuersystem* und *maxime-steuern* in der *jeder-zahlt* Implementierung äquivalent sind.

theorem**fixes** *steuersystem-impl* :: $\langle \text{nat} \Rightarrow \text{nat} \rangle$ **assumes** *steuer-kleiner-einkommen*: $\langle \forall \text{einkommen}. \text{steuersystem-impl } \text{einkommen} \leq \text{einkommen} \rangle$ **and** *existenzminimum*: $\langle \forall \text{einkommen}. \text{einkommen} \leq 9888 \longrightarrow \text{steuersystem-impl } \text{einkommen} = 0 \rangle$ **shows** $\langle (\forall \text{welt}. \text{moralisch } \text{welt } \text{maxime-steuern } (\text{Handlungsabsicht } (\lambda p \text{ w}. \text{Some } (\text{jeder-zahlt } \text{steuersystem-impl } p \text{ w})))) \rangle$ $\longleftrightarrow \text{steuersystem } \text{steuersystem-impl} \rangle$

References

- [1] B. Russell. *Philosophie des Abendlandes — Ihr Zusammenhang mit der politischen und sozialen Entwicklung*. 2012. Aus dem Englischen von Elisabeth Fischer-Wernecke und Ruth Gillischewski, durchgesehen von Rudolf Kaspar.