

A Formalization Of The Dutch Book Argument

Matthew Doty

April 26, 2020

Contents

1	Minimal Logic	2
1.1	Axiomatization	2
1.2	Common Rules	2
1.3	Lists of Assumptions	2
1.3.1	List Implication	2
1.3.2	Definition of Deduction	3
1.3.3	Interpretation as Minimal Logic	3
1.4	The Deduction Theorem	4
1.5	Monotonic Growth in Deductive Power	4
1.6	The Deduction Theorem Revisited	6
1.7	Reflection	7
1.8	The Cut Rule	7
2	Sets of Assumptions	8
2.1	Definition of Deduction	9
2.1.1	Interpretation as Minimal Logic	9
2.2	The Deduction Theorem	10
2.3	Monotonic Growth in Deductive Power	10
2.4	The Deduction Theorem Revisited	10
2.5	Reflection	11
2.6	The Cut Rule	11
2.7	Maximally Consistent Sets For Minimal Logic	12
3	Combinatory Logic	15
3.1	Definitions	15
3.2	Typing	15
3.3	λ -Abstraction	16
3.4	Common Combinators	16
3.5	The Curry Howard Correspondence	17

1 Minimal Logic

```
theory Minimal-Logic
  imports Main
begin
```

This theory presents *minimal logic*, the implicative fragment of intuitionistic logic.

1.1 Axiomatization

Minimal logic is given by the following Hilbert-style axiom system:

```
class Minimal-Logic =
  fixes deduction :: 'a  $\Rightarrow$  bool          ( $\vdash$  - [60] 55)
  fixes implication :: 'a  $\Rightarrow$  'a  $\Rightarrow$  'a    (infixr  $\rightarrow$  70)
  assumes Axiom-1:  $\vdash \varphi \rightarrow \psi \rightarrow \varphi$ 
  assumes Axiom-2:  $\vdash (\varphi \rightarrow \psi \rightarrow \chi) \rightarrow (\varphi \rightarrow \psi) \rightarrow \varphi \rightarrow \chi$ 
  assumes Modus-Ponens:  $\vdash \varphi \rightarrow \psi \Longrightarrow \vdash \varphi \Longrightarrow \vdash \psi$ 
```

A convenience class to have is *Minimal-Logic* extended with a single named constant, intended to be *falsum*. Other classes extending this class will provide rules for how this constant interacts with other terms.

```
class Minimal-Logic-With-Falsum = Minimal-Logic +
  fixes falsum :: 'a                      ( $\perp$ )
```

1.2 Common Rules

```
lemma (in Minimal-Logic) trivial-implication:  $\vdash \varphi \rightarrow \varphi$ 
  by (meson Axiom-1 Axiom-2 Modus-Ponens)
```

```
lemma (in Minimal-Logic) flip-implication:  $\vdash (\varphi \rightarrow \psi \rightarrow \chi) \rightarrow \psi \rightarrow \varphi \rightarrow \chi$ 
  by (meson Axiom-1 Axiom-2 Modus-Ponens)
```

```
lemma (in Minimal-Logic) hypothetical-syllogism:  $\vdash (\psi \rightarrow \chi) \rightarrow (\varphi \rightarrow \psi) \rightarrow \varphi \rightarrow \chi$ 
  by (meson Axiom-1 Axiom-2 Modus-Ponens)
```

```
lemma (in Minimal-Logic) flip-hypothetical-syllogism:
  shows  $\vdash (\psi \rightarrow \varphi) \rightarrow (\varphi \rightarrow \chi) \rightarrow (\psi \rightarrow \chi)$ 
  using Modus-Ponens flip-implication hypothetical-syllogism by blast
```

```
lemma (in Minimal-Logic) implication-absorption:  $\vdash (\varphi \rightarrow \varphi \rightarrow \psi) \rightarrow \varphi \rightarrow \psi$ 
  by (meson Axiom-1 Axiom-2 Modus-Ponens)
```

1.3 Lists of Assumptions

1.3.1 List Implication

Implication given a list of assumptions can be expressed recursively

primrec (in *Minimal-Logic*) *list-implication* :: 'a list \Rightarrow 'a \Rightarrow 'a (**infix** \Rightarrow 80)
where

$\square \Rightarrow \varphi = \varphi$
 $| (\psi \# \Psi) \Rightarrow \varphi = \psi \rightarrow \Psi \Rightarrow \varphi$

1.3.2 Definition of Deduction

Deduction from a list of assumptions can be expressed in terms of (\Rightarrow) .

definition (in *Minimal-Logic*) *list-deduction* :: 'a list \Rightarrow 'a \Rightarrow bool (**infix** \vdash 60)
where

$\Gamma \vdash \varphi \equiv \vdash \Gamma \Rightarrow \varphi$

1.3.3 Interpretation as Minimal Logic

The relation (\vdash) may naturally be interpreted as a *proves* predicate for an instance of minimal logic for a fixed list of assumptions Γ .

Analogues of the two axioms of minimal logic can be naturally stated using list implication.

lemma (in *Minimal-Logic*) *list-implication-Axiom-1*: $\vdash \varphi \rightarrow \Gamma \Rightarrow \varphi$
by (induct Γ , (simp, meson Axiom-1 Axiom-2 Modus-Ponens)+)

lemma (in *Minimal-Logic*) *list-implication-Axiom-2*: $\vdash \Gamma \Rightarrow (\varphi \rightarrow \psi) \rightarrow \Gamma \Rightarrow \varphi$
 $\rightarrow \Gamma \Rightarrow \psi$
by (induct Γ , (simp, meson Axiom-1 Axiom-2 Modus-Ponens hypothetical-syllogism)+)

The lemmas $\vdash ?\varphi \rightarrow ?\Gamma \Rightarrow ?\varphi$ and $\vdash ?\Gamma \Rightarrow (? \varphi \rightarrow ? \psi) \rightarrow ?\Gamma \Rightarrow ? \varphi$
 $\rightarrow ?\Gamma \Rightarrow ? \psi$ jointly give rise to an interpretation of minimal logic, where
a list of assumptions Γ plays the role of a *background theory* of (\vdash) .

context *Minimal-Logic* **begin**

interpretation *List-Deduction-Logic*: *Minimal-Logic* $\lambda \varphi. \Gamma \vdash \varphi (\rightarrow)$

proof qed (meson list-deduction-def

Axiom-1

Axiom-2

Modus-Ponens

list-implication-Axiom-1

list-implication-Axiom-2)+

end

The following *weakening* rule can also be derived.

lemma (in *Minimal-Logic*) *list-deduction-weaken*: $\vdash \varphi \Longrightarrow \Gamma \vdash \varphi$

unfolding *list-deduction-def*

using *Modus-Ponens list-implication-Axiom-1*

by *blast*

In the case of the empty list, the converse may be established.

lemma (in *Minimal-Logic*) *list-deduction-base-theory* [simp]: $\square \vdash \varphi \equiv \vdash \varphi$

unfolding *list-deduction-def*
by *simp*

lemma (in *Minimal-Logic*) *list-deduction-modus-ponens*: $\Gamma \vdash \varphi \rightarrow \psi \implies \Gamma \vdash \varphi \implies \Gamma \vdash \psi$
unfolding *list-deduction-def*
using *Modus-Ponens list-implication-Axiom-2*
by *blast*

1.4 The Deduction Theorem

One result in the meta-theory of minimal logic is the *deduction theorem*, which is a mechanism for moving antecedents back and forth from collections of assumptions.

To develop the deduction theorem, the following two lemmas generalize \vdash ($? \varphi \rightarrow ? \psi \rightarrow ? \chi$) $\rightarrow ? \psi \rightarrow ? \varphi \rightarrow ? \chi$.

lemma (in *Minimal-Logic*) *list-flip-implication1*: $\vdash (\varphi \# \Gamma) \rightarrow \chi \rightarrow \Gamma \rightarrow (\varphi \rightarrow \chi)$
by (*induct* Γ ,
(simp, meson Axiom-1 Axiom-2 Modus-Ponens flip-implication hypothetical-syllogism)+)

lemma (in *Minimal-Logic*) *list-flip-implication2*: $\vdash \Gamma \rightarrow (\varphi \rightarrow \chi) \rightarrow (\varphi \# \Gamma) \rightarrow \chi$
by (*induct* Γ ,
(simp, meson Axiom-1 Axiom-2 Modus-Ponens flip-implication hypothetical-syllogism)+)

Together the two lemmas above suffice to prove a form of the deduction theorem:

theorem (in *Minimal-Logic*) *list-deduction-theorem*: $(\varphi \# \Gamma) \vdash \psi = \Gamma \vdash \varphi \rightarrow \psi$
unfolding *list-deduction-def*
by (*metis Modus-Ponens list-flip-implication1 list-flip-implication2*)

1.5 Monotonic Growth in Deductive Power

In logic, for two sets of assumptions Φ and Ψ , if $\Psi \subseteq \Phi$ then the latter theory Φ is said to be *stronger* than former theory Ψ . In principle, anything a weaker theory can prove a stronger theory can prove. One way of saying this is that deductive power increases monotonically with as the set of underlying assumptions grow.

The monotonic growth of deductive power can be expressed as a meta-theorem in minimal logic.

The lemma $\vdash ? \Gamma \rightarrow (? \varphi \rightarrow ? \chi) \rightarrow (? \varphi \# ? \Gamma) \rightarrow ? \chi$ presents a means of *introducing* assumptions into a list of assumptions when those assumptions have arrived at an implication. The next lemma presents a means of

discharging those assumptions, which can be used in the monotonic growth theorem to be proved.

lemma (in *Minimal-Logic*) *list-implication-removeAll*:
 $\vdash \Gamma \rightarrow \psi \rightarrow (\text{removeAll } \varphi \Gamma) \rightarrow (\varphi \rightarrow \psi)$
proof –
 have $\forall \psi. \vdash \Gamma \rightarrow \psi \rightarrow (\text{removeAll } \varphi \Gamma) \rightarrow (\varphi \rightarrow \psi)$
proof(*induct* Γ)
 case *Nil*
 then show ?case by (*simp, meson Axiom-1*)
next
 case (*Cons* $\chi \Gamma$)
 assume *inductive-hypothesis*: $\forall \psi. \vdash \Gamma \rightarrow \psi \rightarrow \text{removeAll } \varphi \Gamma \rightarrow (\varphi \rightarrow \psi)$
 moreover {
 assume $\varphi \neq \chi$
 with *inductive-hypothesis*
 have $\forall \psi. \vdash (\chi \# \Gamma) \rightarrow \psi \rightarrow \text{removeAll } \varphi (\chi \# \Gamma) \rightarrow (\varphi \rightarrow \psi)$
 by (*simp, meson Modus-Ponens hypothetical-syllogism*)
 }
 moreover {
 fix ψ
 assume $\varphi\text{-equals-}\chi$: $\varphi = \chi$
 moreover with *inductive-hypothesis*
 have $\vdash \Gamma \rightarrow (\chi \rightarrow \psi) \rightarrow \text{removeAll } \varphi (\chi \# \Gamma) \rightarrow (\varphi \rightarrow \chi \rightarrow \psi)$ by *simp*
 hence $\vdash \Gamma \rightarrow (\chi \rightarrow \psi) \rightarrow \text{removeAll } \varphi (\chi \# \Gamma) \rightarrow (\varphi \rightarrow \psi)$
 by (*metis calculation Modus-Ponens implication-absorption list-flip-implication1 list-flip-implication2 list-implication.simps(2)*)
 ultimately have $\vdash (\chi \# \Gamma) \rightarrow \psi \rightarrow \text{removeAll } \varphi (\chi \# \Gamma) \rightarrow (\varphi \rightarrow \psi)$
 by (*simp, metis Modus-Ponens hypothetical-syllogism list-flip-implication1 list-implication.simps(2)*)
 }
 ultimately show ?case by *simp*
qed
 thus ?thesis by *blast*
qed

From lemma above presents what is needed to prove that deductive power for lists is monotonic.

theorem (in *Minimal-Logic*) *list-implication-monotonic*:
 $\text{set } \Sigma \subseteq \text{set } \Gamma \implies \vdash \Sigma \rightarrow \varphi \rightarrow \Gamma \rightarrow \varphi$
proof –
 assume $\text{set } \Sigma \subseteq \text{set } \Gamma$
 moreover have $\forall \Sigma \varphi. \text{set } \Sigma \subseteq \text{set } \Gamma \longrightarrow \vdash \Sigma \rightarrow \varphi \rightarrow \Gamma \rightarrow \varphi$
proof(*induct* Γ)
 case *Nil*
 then show ?case
 by (*metis list-implication.simps(1) list-implication-Axiom-1 set-empty subset-empty*)
next
 case (*Cons* $\psi \Gamma$)
 assume *inductive-hypothesis*: $\forall \Sigma \varphi. \text{set } \Sigma \subseteq \text{set } \Gamma \longrightarrow \vdash \Sigma \rightarrow \varphi \rightarrow \Gamma \rightarrow \varphi$

```

{
  fix  $\Sigma$ 
  fix  $\varphi$ 
  assume  $\Sigma$ -subset-relation:  $\text{set } \Sigma \subseteq \text{set } (\psi \# \Gamma)$ 
  have  $\vdash \Sigma : \rightarrow \varphi \rightarrow (\psi \# \Gamma) : \rightarrow \varphi$ 
  proof -
    {
      assume  $\text{set } \Sigma \subseteq \text{set } \Gamma$ 
      hence ?thesis
        by (metis inductive-hypothesis Axiom-1 Modus-Ponens flip-implication
            list-implication.simps(2))
    }
    moreover {
      let  $? \Delta = \text{removeAll } \psi \Sigma$ 
      assume  $\sim (\text{set } \Sigma \subseteq \text{set } \Gamma)$ 
      hence  $\text{set } ? \Delta \subseteq \text{set } \Gamma$  using  $\Sigma$ -subset-relation by auto
      hence  $\vdash ? \Delta : \rightarrow (\psi \rightarrow \varphi) \rightarrow \Gamma : \rightarrow (\psi \rightarrow \varphi)$  using inductive-hypothesis
    by auto
      hence  $\vdash ? \Delta : \rightarrow (\psi \rightarrow \varphi) \rightarrow (\psi \# \Gamma) : \rightarrow \varphi$ 
      by (metis Modus-Ponens
          flip-implication
          list-flip-implication2
          list-implication.simps(2))
      moreover have  $\vdash \Sigma : \rightarrow \varphi \rightarrow ? \Delta : \rightarrow (\psi \rightarrow \varphi)$ 
      by (simp add: local.list-implication-removeAll)
      ultimately have ?thesis
        using Modus-Ponens hypothetical-syllogism by blast
    }
    ultimately show ?thesis by blast
  qed
}
thus ?case by simp
qed
ultimately show ?thesis by simp
qed

```

A direct consequence is that deduction from lists of assumptions is monotonic as well:

theorem (in *Minimal-Logic*) *list-deduction-monotonic*:
 $\text{set } \Sigma \subseteq \text{set } \Gamma \implies \Sigma : \vdash \varphi \implies \Gamma : \vdash \varphi$
unfolding *list-deduction-def*
using *Modus-Ponens list-implication-monotonic*
by *blast*

1.6 The Deduction Theorem Revisited

The monotonic nature of deduction allows us to prove another form of the deduction theorem, where the assumption being discharged is completely removed from the list of assumptions.

theorem (in *Minimal-Logic*) *alternate-list-deduction-theorem*:

$(\varphi \# \Gamma) \vdash \psi = (\text{removeAll } \varphi \ \Gamma) \vdash \varphi \rightarrow \psi$

by (*metis list-deduction-def*
Modus-Ponens
filter-is-subset
list-deduction-monotonic
list-deduction-theorem
list-implication-removeAll
removeAll.simps(2)
removeAll-filter-not-eq)

1.7 Reflection

In logic the *reflection* principle sometimes refers to when a collection of assumptions can deduce any of its members. It is automatically derivable from $\llbracket \text{set } ?\Sigma \subseteq \text{set } ?\Gamma; ?\Sigma \vdash ?\varphi \rrbracket \implies ?\Gamma \vdash ?\varphi$ among the other rules provided.

lemma (in *Minimal-Logic*) *list-deduction-reflection*: $\varphi \in \text{set } \Gamma \implies \Gamma \vdash \varphi$

by (*metis list-deduction-def*
insert-subset
list.simps(15)
list-deduction-monotonic
list-implication.simps(2)
list-implication-Axiom-1
order-refl)

1.8 The Cut Rule

Cut is a rule commonly presented in sequent calculi, dating back to Gerhard Gentzen's "Investigations in Logical Deduction" (1934) TODO: Cite me

The cut rule is not generally necessary in sequent calculi. It can often be shown that the rule can be eliminated without reducing the power of the underlying logic. However, as demonstrated by George Boolos' *Don't Eliminate Cute* (1984) (TODO: Cite me), removing the rule can often lead to very inefficient proof systems.

Here the rule is presented just as a meta theorem.

theorem (in *Minimal-Logic*) *list-deduction-cut-rule*:

$(\varphi \# \Gamma) \vdash \psi \implies \Delta \vdash \varphi \implies \Gamma @ \Delta \vdash \psi$

by (*metis*
(no-types, lifting)
Un-upper1
Un-upper2
list-deduction-modus-ponens
list-deduction-monotonic
list-deduction-theorem)

set-append)

The cut rule can also be strengthened to entire lists of propositions.

theorem (in *Minimal-Logic*) *strong-list-deduction-cut-rule*:

$(\Phi @ \Gamma) : \vdash \psi \implies \forall \varphi \in \text{set } \Phi. \Delta : \vdash \varphi \implies \Gamma @ \Delta : \vdash \psi$

proof –

have $\forall \psi. (\Phi @ \Gamma : \vdash \psi \longrightarrow (\forall \varphi \in \text{set } \Phi. \Delta : \vdash \varphi) \longrightarrow \Gamma @ \Delta : \vdash \psi)$

proof(*induct* Φ)

case *Nil*

then show *?case*

by (*metis Un-iff append.left-neutral list-deduction-monotonic set-append*

subsetI)

next

case (*Cons* $\chi \Phi$)

assume *inductive-hypothesis*:

$\forall \psi. \Phi @ \Gamma : \vdash \psi \longrightarrow (\forall \varphi \in \text{set } \Phi. \Delta : \vdash \varphi) \longrightarrow \Gamma @ \Delta : \vdash \psi$

{

fix $\psi \chi$

assume $(\chi \# \Phi) @ \Gamma : \vdash \psi$

hence $A: \Phi @ \Gamma : \vdash \chi \rightarrow \psi$ **using** *list-deduction-theorem* **by** *auto*

assume $\forall \varphi \in \text{set } (\chi \# \Phi). \Delta : \vdash \varphi$

hence $B: \forall \varphi \in \text{set } \Phi. \Delta : \vdash \varphi$

and $C: \Delta : \vdash \chi$ **by** *auto*

from $A B$ **have** $\Gamma @ \Delta : \vdash \chi \rightarrow \psi$ **using** *inductive-hypothesis* **by** *blast*

with C **have** $\Gamma @ \Delta : \vdash \psi$

by (*meson list.set-intros(1)*

list-deduction-cut-rule

list-deduction-modus-ponens

list-deduction-reflection)

}

thus *?case* **by** *simp*

qed

moreover assume $(\Phi @ \Gamma) : \vdash \psi$

moreover assume $\forall \varphi \in \text{set } \Phi. \Delta : \vdash \varphi$

ultimately show *?thesis* **by** *blast*

qed

2 Sets of Assumptions

While deduction in terms of lists of assumptions is straight-forward to define, deduction (and the *deduction theorem*) is commonly given in terms of *sets* of propositions. This formulation is suited to establishing strong completeness theorems and compactness theorems.

The presentation of deduction from a set follows the presentation of list deduction given for $(: \vdash)$.

2.1 Definition of Deduction

Just as deduction from a list (\vdash) can be defined in terms of (\rightarrow), deduction from a *set* of assumptions can be expressed in terms of (\vdash).

definition (in *Minimal-Logic*) *set-deduction* :: 'a set \Rightarrow 'a \Rightarrow bool (infix \Vdash 60)
where

$$\Gamma \Vdash \varphi \equiv \exists \Psi. \text{set}(\Psi) \subseteq \Gamma \wedge \Psi \vdash \varphi$$

2.1.1 Interpretation as Minimal Logic

As in the case of (\vdash), the relation (\Vdash) may be interpreted as a *proves* predicate for a fixed set of assumptions Γ .

The following lemma is given in order to establish this, which asserts that every minimal logic tautology $\vdash \varphi$ is also a tautology for $\Gamma \Vdash \varphi$.

lemma (in *Minimal-Logic*) *set-deduction-weaken*: $\vdash \varphi \implies \Gamma \Vdash \varphi$
using *list-deduction-base-theory set-deduction-def* by *fastforce*

In the case of the empty set, the converse may be established.

lemma (in *Minimal-Logic*) *set-deduction-base-theory*: $\{\} \Vdash \varphi \equiv \vdash \varphi$
using *list-deduction-base-theory set-deduction-def* by *auto*

Next, a form of *modus ponens* is provided for (\Vdash).

lemma (in *Minimal-Logic*) *set-deduction-modus-ponens*: $\Gamma \Vdash \varphi \rightarrow \psi \implies \Gamma \Vdash \varphi \implies \Gamma \Vdash \psi$

proof –

assume $\Gamma \Vdash \varphi \rightarrow \psi$
then obtain Φ where A : set $\Phi \subseteq \Gamma$ and B : $\Phi \vdash \varphi \rightarrow \psi$
using *set-deduction-def* by *blast*
assume $\Gamma \Vdash \varphi$
then obtain Ψ where C : set $\Psi \subseteq \Gamma$ and D : $\Psi \vdash \varphi$
using *set-deduction-def* by *blast*
from B D have $\Phi @ \Psi \vdash \psi$
using *list-deduction-cut-rule list-deduction-theorem* by *blast*
moreover from A C have set $(\Phi @ \Psi) \subseteq \Gamma$ by *simp*
ultimately show *?thesis*
using *set-deduction-def* by *blast*

qed

context *Minimal-Logic* **begin**

interpretation *Set-Deduction-Logic*: *Minimal-Logic* $\lambda \varphi. \Gamma \Vdash \varphi (\rightarrow)$

proof

fix $\varphi \psi$
show $\Gamma \Vdash \varphi \rightarrow \psi \rightarrow \varphi$ by (*metis Axiom-1 set-deduction-weaken*)

next

fix $\varphi \psi \chi$
show $\Gamma \Vdash (\varphi \rightarrow \psi \rightarrow \chi) \rightarrow (\varphi \rightarrow \psi) \rightarrow \varphi \rightarrow \chi$ by (*metis Axiom-2 set-deduction-weaken*)

```

next
  fix  $\varphi \ \psi$ 
  show  $\Gamma \Vdash \varphi \rightarrow \psi \implies \Gamma \Vdash \varphi \implies \Gamma \Vdash \psi$  using set-deduction-modus-ponens by
metis
qed
end

```

2.2 The Deduction Theorem

The next result gives the deduction theorem for (\Vdash) .

```

theorem (in Minimal-Logic) set-deduction-theorem:  $\text{insert } \varphi \ \Gamma \Vdash \psi = \Gamma \Vdash \varphi \rightarrow \psi$ 
proof –
  have  $\Gamma \Vdash \varphi \rightarrow \psi \implies \text{insert } \varphi \ \Gamma \Vdash \psi$ 
    by (metis set-deduction-def insert-mono list.simps(15) list-deduction-theorem)
  moreover {
    assume  $\text{insert } \varphi \ \Gamma \Vdash \psi$ 
    then obtain  $\Phi$  where  $\text{set } \Phi \subseteq \text{insert } \varphi \ \Gamma$  and  $\Phi \vdash \psi$ 
      using set-deduction-def by auto
    hence  $\text{set } (\text{removeAll } \varphi \ \Phi) \subseteq \Gamma$  by auto
    moreover from  $\langle \Phi \vdash \psi \rangle$  have  $\text{removeAll } \varphi \ \Phi \vdash \varphi \rightarrow \psi$ 
      using Modus-Ponens list-implication-removeAll list-deduction-def
      by blast
    ultimately have  $\Gamma \Vdash \varphi \rightarrow \psi$ 
      using set-deduction-def by blast
  }
  ultimately show  $\text{insert } \varphi \ \Gamma \Vdash \psi = \Gamma \Vdash \varphi \rightarrow \psi$  by metis
qed

```

2.3 Monotonic Growth in Deductive Power

In contrast to the (\vdash) relation, the proof that the deductive power of (\Vdash) grows monotonically with its assumptions may be fully automated.

```

theorem set-deduction-monotonic:  $\Sigma \subseteq \Gamma \implies \Sigma \vdash \varphi \implies \Gamma \Vdash \varphi$ 
by (meson dual-order.trans set-deduction-def)

```

2.4 The Deduction Theorem Revisited

As a consequence of the fact that $\llbracket ?\Sigma \subseteq ?\Gamma; ?\Sigma \vdash ?\varphi \rrbracket \implies ?\Gamma \Vdash ?\varphi$ automatically provable, the alternate *deduction theorem* where the discharged assumption is completely removed from the set of assumptions is just a consequence of the more conventional $\text{insert } ?\varphi \ ?\Gamma \Vdash ?\psi = ?\Gamma \Vdash ?\varphi \rightarrow ?\psi$ and some basic set identities.

```

theorem (in Minimal-Logic) alternate-set-deduction-theorem:
   $\text{insert } \varphi \ \Gamma \Vdash \psi = \Gamma - \{\varphi\} \Vdash \varphi \rightarrow \psi$ 
by (metis insert-Diff-single set-deduction-theorem)

```

2.5 Reflection

Just as in the case of $(:\vdash)$, deduction from sets of assumptions makes true the *reflection principle* and is automatically provable.

theorem (in *Minimal-Logic*) *set-deduction-reflection*: $\varphi \in \Gamma \implies \Gamma \Vdash \varphi$
by (*metis Set.set-insert*
list-implication.simps(1)
list-implication-Axiom-1
set-deduction-theorem
set-deduction-weaken)

2.6 The Cut Rule

The final principle of (\Vdash) presented is the *cut rule*.

First, the weak form of the rule is established.

theorem (in *Minimal-Logic*) *set-deduction-cut-rule*:
 $\text{insert } \varphi \Gamma \Vdash \psi \implies \Delta \Vdash \varphi \implies \Gamma \cup \Delta \Vdash \psi$
proof –
assume $\text{insert } \varphi \Gamma \Vdash \psi$
hence $\Gamma \Vdash \varphi \rightarrow \psi$ **using** *set-deduction-theorem* **by** *auto*
hence $\Gamma \cup \Delta \Vdash \varphi \rightarrow \psi$ **using** *set-deduction-def* **by** *auto*
moreover **assume** $\Delta \Vdash \varphi$
hence $\Gamma \cup \Delta \Vdash \varphi$ **using** *set-deduction-def* **by** *auto*
ultimately show *?thesis* **using** *set-deduction-modus-ponens* **by** *metis*
qed

Another lemma is shown next in order to establish the strong form of the cut rule. The lemma shows the existence of a *covering list* of assumptions Ψ in the event some set of assumptions Δ proves everything in a finite set of assumptions Φ .

lemma (in *Minimal-Logic*) *finite-set-deduction-list-deduction*:
 $\text{finite } \Phi \implies$
 $\forall \varphi \in \Phi. \Delta \Vdash \varphi \implies$
 $\exists \Psi. \text{set } \Psi \subseteq \Delta \wedge (\forall \varphi \in \Phi. \Psi \vdash \varphi)$
proof(*induct* Φ *rule: finite-induct*)
case *empty* **thus** *?case* **by** (*metis all-not-in-conv empty-subsetI set-empty*)
next
case (*insert* χ Φ)
assume $\forall \varphi \in \Phi. \Delta \Vdash \varphi \implies \exists \Psi. \text{set } \Psi \subseteq \Delta \wedge (\forall \varphi \in \Phi. \Psi \vdash \varphi)$
and $\forall \varphi \in \text{insert } \chi \Phi. \Delta \Vdash \varphi$
hence $\exists \Psi. \text{set } \Psi \subseteq \Delta \wedge (\forall \varphi \in \Phi. \Psi \vdash \varphi)$ **and** $\Delta \Vdash \chi$ **by** *simp+*
then obtain $\Psi_1 \Psi_2$ **where**
 $\text{set } (\Psi_1 @ \Psi_2) \subseteq \Delta$ **and**
 $\forall \varphi \in \Phi. \Psi_1 \vdash \varphi$ **and**
 $\Psi_2 \vdash \chi$
using *set-deduction-def* **by** *auto*
moreover from this have $\forall \varphi \in (\text{insert } \chi \Phi). \Psi_1 @ \Psi_2 \vdash \varphi$

```

    by (metis
        insert-iff
        le-sup-iff
        list-deduction-monotonic
        order-refl set-append)
    ultimately show ?case by blast
qed

```

With $\llbracket \text{finite } ?\Phi; \forall \varphi \in ?\Phi. ?\Delta \Vdash \varphi \rrbracket \implies \exists \Psi. \text{set } \Psi \subseteq ?\Delta \wedge (\forall \varphi \in ?\Phi. \Psi \vdash \varphi)$ the strengthened form of the cut rule can be given.

theorem (in *Minimal-Logic*) *strong-set-deduction-cut-rule*:

$\Phi \cup \Gamma \Vdash \psi \implies \forall \varphi \in \Phi. \Delta \Vdash \varphi \implies \Gamma \cup \Delta \Vdash \psi$

proof –

assume $\Phi \cup \Gamma \Vdash \psi$

then obtain Σ where

A : $\text{set } \Sigma \subseteq \Phi \cup \Gamma$ and

B : $\Sigma \vdash \psi$

using *set-deduction-def*

by *auto*+

obtain $\Phi' \Gamma'$ where

C : $\text{set } \Phi' = \text{set } \Sigma \cap \Phi$ and

D : $\text{set } \Gamma' = \text{set } \Sigma \cap \Gamma$

by (metis *inf-sup-aci*(1) *inter-set-filter*)+

then have $\text{set } (\Phi' @ \Gamma') = \text{set } \Sigma$ using A by *auto*

hence E : $\Phi' @ \Gamma' \vdash \psi$ using B *list-deduction-monotonic* by *blast*

assume $\forall \varphi \in \Phi. \Delta \Vdash \varphi$

hence $\forall \varphi \in \text{set } \Phi'. \Delta \Vdash \varphi$ using C by *auto*

from this obtain Δ' where $\text{set } \Delta' \subseteq \Delta$ and $\forall \varphi \in \text{set } \Phi'. \Delta' \vdash \varphi$

using *finite-set-deduction-list-deduction* by *blast*

with *strong-list-deduction-cut-rule* D E

have $\text{set } (\Gamma' @ \Delta') \subseteq \Gamma \cup \Delta$ and $\Gamma' @ \Delta' \vdash \psi$ by *auto*

thus *?thesis* using *set-deduction-def* by *blast*

qed

2.7 Maximally Consistent Sets For Minimal Logic

definition (in *Minimal-Logic*)

Formula-Consistent :: 'a \Rightarrow 'a *set* \Rightarrow bool (*--Consistent* - [100] 100)

where [*simp*]: $\varphi\text{-Consistent } \Gamma \equiv \sim (\Gamma \Vdash \varphi)$

lemma (in *Minimal-Logic*) *Formula-Consistent-Extension*:

assumes $\varphi\text{-Consistent } \Gamma$

shows $(\varphi\text{-Consistent insert } \psi \Gamma) \vee (\varphi\text{-Consistent insert } (\psi \rightarrow \varphi) \Gamma)$

proof –

{

assume $\sim \varphi\text{-Consistent insert } \psi \Gamma$

hence $\Gamma \Vdash \psi \rightarrow \varphi$

using *set-deduction-theorem*

unfolding *Formula-Consistent-def*

```

    by simp
  hence  $\varphi$ -Consistent insert  $(\psi \rightarrow \varphi)$   $\Gamma$ 
    by (metis Un-absorb assms Formula-Consistent-def set-deduction-cut-rule)
}
thus ?thesis by blast
qed

```

definition (in *Minimal-Logic*)

Formula-Maximally-Consistent-Set

$:: 'a \Rightarrow 'a \text{ set} \Rightarrow \text{bool}$ ($\neg \text{MCS} - [100] 100$)

where

[simp]: $\varphi\text{-MCS } \Gamma \equiv (\varphi\text{-Consistent } \Gamma) \wedge (\forall \psi. \psi \in \Gamma \vee (\psi \rightarrow \varphi) \in \Gamma)$

theorem (in *Minimal-Logic*) *Formula-Maximally-Consistent-Extension*:

assumes $\varphi\text{-Consistent } \Gamma$

shows $\exists \Omega. (\varphi\text{-MCS } \Omega) \wedge \Gamma \subseteq \Omega$

proof –

let $? \Gamma\text{-Extensions} = \{\Sigma. (\varphi\text{-Consistent } \Sigma) \wedge \Gamma \subseteq \Sigma\}$

have $\exists \Omega \in ? \Gamma\text{-Extensions}. \forall \Sigma \in ? \Gamma\text{-Extensions}. \Omega \subseteq \Sigma \longrightarrow \Sigma = \Omega$

proof (rule subset-Zorn)

fix $\mathcal{C} :: 'a \text{ set set}$

assume subset-chain- \mathcal{C} : subset.chain $? \Gamma\text{-Extensions } \mathcal{C}$

hence \mathcal{C} : $\forall \Sigma \in \mathcal{C}. \Gamma \subseteq \Sigma \wedge \forall \Sigma \in \mathcal{C}. \varphi\text{-Consistent } \Sigma$

unfolding subset.chain-def by blast+

show $\exists \Omega \in ? \Gamma\text{-Extensions}. \forall \Sigma \in \mathcal{C}. \Sigma \subseteq \Omega$

proof cases

assume $\mathcal{C} = \{\}$ thus ?thesis using assms by blast

next

let $? \Omega = \bigcup \mathcal{C}$

assume $\mathcal{C} \neq \{\}$

hence $\Gamma \subseteq ? \Omega$ by (simp add: $\mathcal{C}(1)$ less-eq-Sup)

moreover have $\varphi\text{-Consistent } ? \Omega$

proof –

{

assume $\sim \varphi\text{-Consistent } ? \Omega$

then obtain ω where ω : finite $\omega \subseteq ? \Omega \sim \varphi\text{-Consistent } \omega$

unfolding Formula-Consistent-def
set-deduction-def

by auto

from $\omega(1) \ \omega(2)$ have $\exists \Sigma \in \mathcal{C}. \omega \subseteq \Sigma$

proof (induct ω rule: finite-induct)

case empty thus ?case using $\mathcal{C} \neq \{\}$ by blast

next

case (insert $\psi \ \omega$)

from this obtain $\Sigma_1 \ \Sigma_2$ where

Σ_1 : $\omega \subseteq \Sigma_1 \ \Sigma_1 \in \mathcal{C}$ and

Σ_2 : $\psi \in \Sigma_2 \ \Sigma_2 \in \mathcal{C}$

by auto

hence $\Sigma_1 \subseteq \Sigma_2 \vee \Sigma_2 \subseteq \Sigma_1$

```

      using subset-chain- $\mathcal{C}$ 
      unfolding subset.chain-def
      by blast
    hence  $(\text{insert } \psi \ \omega) \subseteq \Sigma_1 \vee (\text{insert } \psi \ \omega) \subseteq \Sigma_2$ 
      using  $\Sigma_1 \ \Sigma_2$  by blast
    thus ?case using  $\Sigma_1 \ \Sigma_2$  by blast
  qed
  hence  $\exists \Sigma \in \mathcal{C}. (\varphi\text{-Consistent } \Sigma) \wedge \sim (\varphi\text{-Consistent } \Sigma)$ 
    using  $\mathcal{C}(2) \ \omega(3)$ 
    unfolding Formula-Consistent-def
      set-deduction-def
    by auto
  hence False by auto
}
thus ?thesis by blast
qed
ultimately show ?thesis by blast
qed
then obtain  $\Omega$  where  $\Omega: \Omega \in ?\Gamma\text{-Extensions}$ 
   $\forall \Sigma \in ?\Gamma\text{-Extensions}. \Omega \subseteq \Sigma \longrightarrow \Sigma = \Omega$  by auto+
{
  fix  $\psi$ 
  have  $(\varphi\text{-Consistent } \text{insert } \psi \ \Omega) \vee (\varphi\text{-Consistent } \text{insert } (\psi \rightarrow \varphi) \ \Omega)$ 
     $\Gamma \subseteq \text{insert } \psi \ \Omega$ 
     $\Gamma \subseteq \text{insert } (\psi \rightarrow \varphi) \ \Omega$ 
    using  $\Omega(1)$  Formula-Consistent-Extension Formula-Consistent-def
    by auto
  hence  $\text{insert } \psi \ \Omega \in ?\Gamma\text{-Extensions}$ 
     $\vee \text{insert } (\psi \rightarrow \varphi) \ \Omega \in ?\Gamma\text{-Extensions}$ 
    by blast
  hence  $\psi \in \Omega \vee (\psi \rightarrow \varphi) \in \Omega$  using  $\Omega(2)$  by blast
}
thus ?thesis
  using  $\Omega(1)$ 
  unfolding Formula-Maximally-Consistent-Set-def
  by blast
qed

lemma (in Minimal-Logic) Formula-Maximally-Consistent-Set-reflection:
   $\varphi\text{-MCS } \Gamma \implies \psi \in \Gamma = \Gamma \Vdash \psi$ 
proof -
  assume  $\varphi\text{-MCS } \Gamma$ 
  {
    assume  $\Gamma \Vdash \psi$ 
    moreover from  $\langle \varphi\text{-MCS } \Gamma \rangle$  have  $\psi \in \Gamma \vee (\psi \rightarrow \varphi) \in \Gamma \sim \Gamma \Vdash \varphi$ 
      unfolding Formula-Maximally-Consistent-Set-def Formula-Consistent-def
      by auto
    ultimately have  $\psi \in \Gamma$ 

```

```

    using set-deduction-reflection set-deduction-modus-ponens
    by metis
  }
  thus  $\psi \in \Gamma = \Gamma \vdash \psi$ 
    using set-deduction-reflection
    by metis
qed

theorem (in Minimal-Logic)
  Formula-Maximally-Consistent-Set-implication-elimination:
  assumes  $\varphi \text{--}MCS \ \Omega$ 
  shows  $(\psi \rightarrow \chi) \in \Omega \implies \psi \in \Omega \implies \chi \in \Omega$ 
  using assms
    Formula-Maximally-Consistent-Set-reflection
    set-deduction-modus-ponens
  by blast

end

```

3 Combinatory Logic

```

theory Combinators
  imports ./Minimal-Logic
begin

```

3.1 Definitions

Combinatory logic, following Curry (TODO: cite me), can be formulated as follows.

```

datatype Var = Var nat ( $\mathcal{X}$ )

datatype SKCombinator =
  Variable-Combinator Var ( $\langle \cdot \rangle$  [100] 100)
| S-Combinator (S)
| K-Combinator (K)
| Combinator-Application SKCombinator SKCombinator (infixl · 75)

```

Note that in addition to S and K combinators, $SKCombinator$ provides terms for *variables*. This is helpful when studying λ -abstraction embedding.

3.2 Typing

The fragment of the $SKCombinator$ types without *Variable-Combinator* terms can be given *simple types*:

```

datatype 'a Simple-Type =
  Atom 'a ( $\mathbb{A} - \mathbb{B}$  [100] 100)
| To 'a Simple-Type 'a Simple-Type (infixr  $\Rightarrow$  70)

```

inductive

Simply-Typed-SKCombinator

$:: SKCombinator \Rightarrow 'a \text{ Simple-Type} \Rightarrow bool$ (**infix** $:: 65$)

where

$S\text{-type} : S :: (\varphi \Rightarrow \psi \Rightarrow \chi) \Rightarrow (\varphi \Rightarrow \psi) \Rightarrow \varphi \Rightarrow \chi$

$| K\text{-type} : K :: \varphi \Rightarrow \psi \Rightarrow \varphi$

$| Application\text{-type} : E_1 :: \varphi \Rightarrow \psi \Longrightarrow E_2 :: \varphi \Longrightarrow E_1 \cdot E_2 :: \psi$

3.3 λ -Abstraction

Here a simple embedding of the λ -calculus into combinator logic is presented.

The SKI embedding below is originally due to David Turner [1].

Abstraction over combinators where the abstracted variable is not free are simplified using the K combinator.

primrec *free-variables-in-SKCombinator* $:: SKCombinator \Rightarrow Var \text{ set } (free_{SK})$

where

$free_{SK} (\langle x \rangle) = \{x\}$

$| free_{SK} S = \{\}$

$| free_{SK} K = \{\}$

$| free_{SK} (E_1 \cdot E_2) = (free_{SK} E_1) \cup (free_{SK} E_2)$

primrec *Turner-Abstraction*

$:: Var \Rightarrow SKCombinator \Rightarrow SKCombinator$ ($\lambda \text{ } - . - [90,90] \text{ } 90$)

where

abst-app:

$\lambda x. (E_1 \cdot E_2) = (if (x \in free_{SK} (E_1 \cdot E_2))$
 $\quad then S \cdot (\lambda x. E_1) \cdot (\lambda x. E_2)$
 $\quad else K \cdot (E_1 \cdot E_2))$

$| abst\text{-}S: \lambda x. S = K \cdot S$

$| abst\text{-}K: \lambda x. K = K \cdot K$

$| abst\text{-}var: \lambda x. \langle y \rangle = (if x = y then S \cdot K \cdot K else K \cdot \langle y \rangle)$

3.4 Common Combinators

This section presents various common combinators. Some combinators are simple enough to express in using S and K , however others are more easily expressed using λ -abstraction. TODO: Cite Haskell Curry's PhD thesis.

A useful lemma is the type of the *identity* combinator, designated by I in the literature.

lemma *Identity-type*: $S \cdot K \cdot K :: \varphi \Rightarrow \varphi$

using $K\text{-type}$ $S\text{-type}$ $Application\text{-type}$ **by** *blast*

Another significant combinator is the combinator, which corresponds to **flip** in Haskell.

lemma *C-type*:

$\lambda \mathcal{X} \ 1. \ \lambda \mathcal{X} \ 2. \ \lambda \mathcal{X} \ 3. \ (\langle \mathcal{X} \ 1 \rangle \cdot \langle \mathcal{X} \ 3 \rangle \cdot \langle \mathcal{X} \ 2 \rangle)$
 $:: (\varphi \Rightarrow \psi \Rightarrow \chi) \Rightarrow \psi \Rightarrow \varphi \Rightarrow \chi$
by (*simp*, *meson Identity-type Simply-Typed-SKCombinator.simps*)

Haskell also has a function $(.)$, which is referred to as the B combinator.

lemma *B-type*: $S \cdot (K \cdot S) \cdot K :: (\psi \Rightarrow \chi) \Rightarrow (\varphi \Rightarrow \psi) \Rightarrow \varphi \Rightarrow \chi$
by (*meson Simply-Typed-SKCombinator.simps*)

The final combinator given is the B combinator.

lemma *W-type*:

$\lambda \mathcal{X} \ 1. \ \lambda \mathcal{X} \ 2. \ (\langle \mathcal{X} \ 1 \rangle \cdot \langle \mathcal{X} \ 2 \rangle \cdot \langle \mathcal{X} \ 2 \rangle) :: (\varphi \Rightarrow \varphi \Rightarrow \chi) \Rightarrow \varphi \Rightarrow \chi$
by (*simp*, *meson Identity-type Simply-Typed-SKCombinator.simps*)

3.5 The Curry Howard Correspondence

The (polymorphic) typing for a combinator X is given by the relation $X :: \varphi$.

Combinator types form an instance of minimal logic.

interpretation *Combinator-Minimal-Logic*: *Minimal-Logic* $\lambda \varphi. \exists X. X :: \varphi (\Rightarrow)$
proof qed (*meson Simply-Typed-SKCombinator.intros*) $+$

The minimal logic generated by combinator logic is *free* in the following sense: If $X :: \varphi$ holds for some combinator X then φ may be interpreted as logical consequence in any given minimal logic instance.

The fact that any valid type in combinator logic may be interpreted in minimal logic is a form of the *Curry-Howard correspondence*. TODO: Cite

primrec (**in** *Minimal-Logic*) *Simple-Type-interpretation*
 $:: 'a \text{ Simple-Type} \Rightarrow 'a \ (\llbracket - \rrbracket \ [50])$ **where**
 $\llbracket \text{Atom } p \rrbracket = p$
 $\llbracket \varphi \Rightarrow \psi \rrbracket = \llbracket \varphi \rrbracket \rightarrow \llbracket \psi \rrbracket$

lemma (**in** *Minimal-Logic*) *Curry-Howard-correspondence*:

$X :: \varphi \Longrightarrow \vdash \llbracket \varphi \rrbracket$
by (*induct rule: Simply-Typed-SKCombinator.induct*,
(simp add: Axiom-1 Axiom-2 Modus-Ponens)) $+$

end

References

- [1] D. A. Turner. Another algorithm for bracket abstraction. *Journal of Symbolic Logic*, 44(2):267–270, June 1979.