Interactive Higher Order Theorem Proving on the Web

Chad E. Brown

cebrown@ps.uni-sb.de

Universität des Saarlandes

Motivation

Want an Interactive Theorem Prover...

- Requiring No Installation.
- Supporting Simple Type Theory/Higher Order Logic.
- That's Easy to Learn and Easy to Use.
- **Students can Use to do Proofs.**

JavaScript Interactive Higher Order Theorem Prover

- Runs in a Web Browser
- Grammar for Language fits on One Page

A Simple Propositional Example

Three Propositional Constants: x, z, and u.
Assume two clauses: $\neg x \vee z$ and $x \vee u$.
Prove $z \vee u$.

Specification of the problem in the prover:

```
const x z u:B
axiom ~x|z
axiom x|u
claim z|u
```

Prove

Simple Example

This is the only open branch.

Import Axiom or Lemma

Special Symbols and ASCII equivalents: ¬ (~) ∨ (|) ∧ (&) → (->) ↔ (<->) ∀ (!) ∃ (?) λ (\) ≠ (!=) Grammar Help Page

---

¬(z∨u) Negation Normalize DeMorgan Delete

Extend

Simple Example

This is the only open branch.

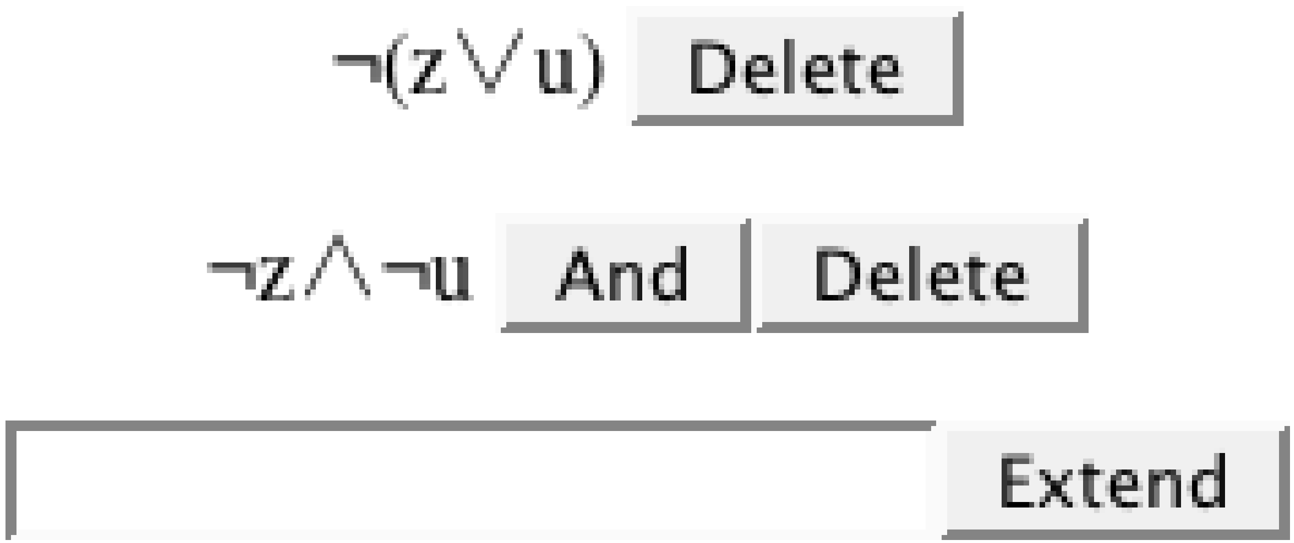
Import Axiom or Lemma

Special Symbols and ASCII equivalents: ¬ (~) ∨ (|) ∧ (&) → (->) ↔ (<->) ∀ (!) ∃ (?) λ (\) ≠ (!=) Grammar Help Page

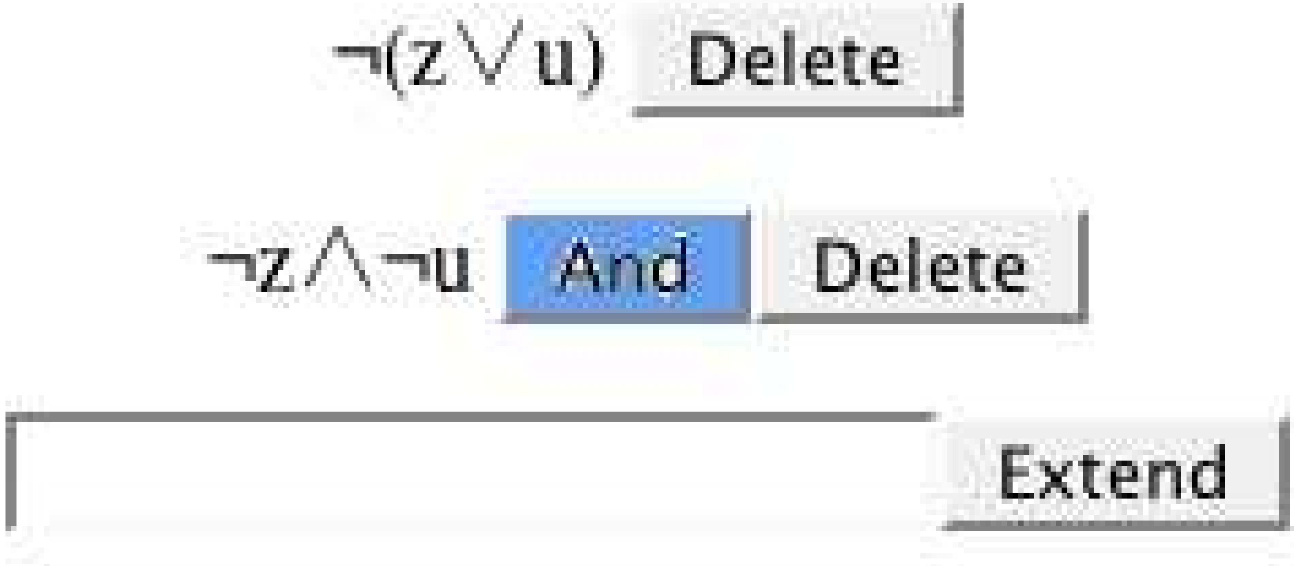
$\neg(z \vee u)$ Negation Normalize DeMorgan Delete

Extend

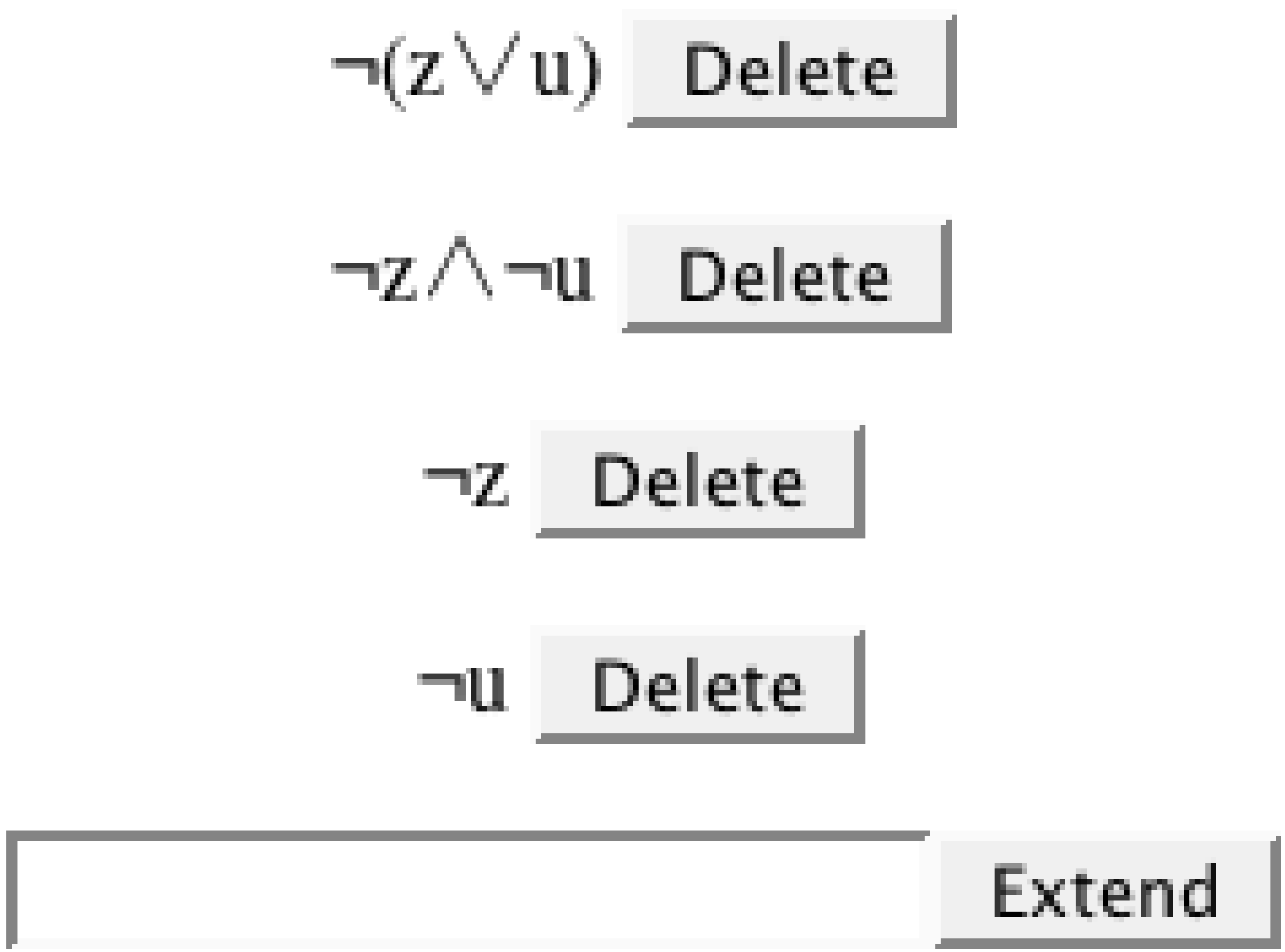
Simple Example



Simple Example



Simple Example



Simple Example

This is the only open branch.

Import Axiom or Lemma | Undo

Special Symbols and ASCII equivalents: ¬ (~) ∨ (|) ∧ (&) → (->) ↔ (<->) ∀ (!) ∃ (?) λ (\) ≠ (!=) Grammar Help Page

---

¬(z∨u) Delete

¬z∧¬u Delete

¬z Delete

¬u Delete

Extend

Simple Example

This is the only open branch.

Import Axiom or Lemma | Undo

Special Symbols and ASCII equivalents: ¬ (~) ∨ (|) ∧ (&) → (->) ↔ (<->) ∀ (!) ∃ (?) λ (\) ≠ (!=) Grammar Help Page

¬(z∨u) Delete

¬z∧¬u Delete

¬z Delete

¬u Delete

Extend

Simple Example

This is the only open branch.

Axiom 1 $\neg x \vee z$

Axiom 2 $x \vee u$

Undo

Special Symbols and ASCII equivalents: ¬ (~) ∨ (|) ∧ (&) → (->) ↔ (<->) ∀ (!) ∃ (?) λ (\) ≠ (!=) Grammar Help Page

---

$\neg(z \vee u)$ Delete

$\neg z \wedge \neg u$ Delete

$\neg z$ Delete

$\neg u$ Delete

Extend

Simple Example

$\neg(z \vee u)$ Delete

$\neg z \wedge \neg u$ Delete

$\neg z$ Delete

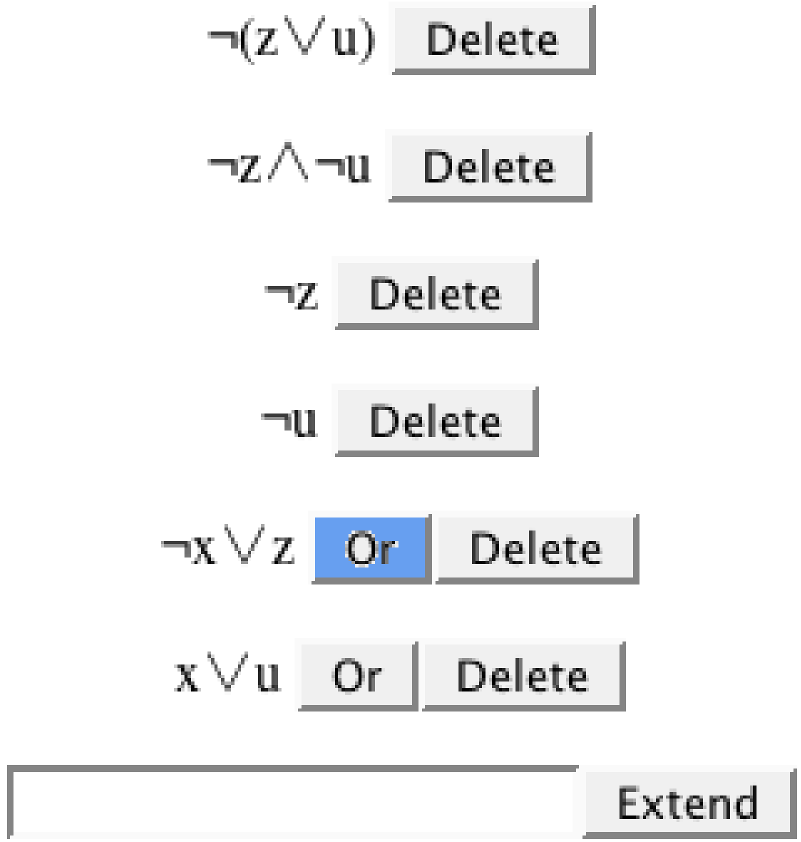
$\neg u$ Delete

$\neg x \vee z$ Or Delete

$x \vee u$ Or Delete

Extend

Simple Example



Simple Example

Simple Example

$\neg(z \vee u)$ Delete

$\neg z \wedge \neg u$ Delete

$\neg z$ Delete

$\neg u$ Delete

$\neg x \vee z$ Delete

$x \vee u$ Or Delete

$\neg x$ Delete

Extend

Simple Example

Complete Proof!

$\neg(z \vee u)$

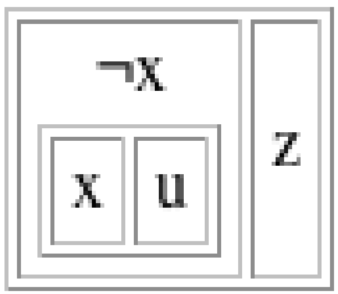
$\neg z \wedge \neg u$

$\neg z$

$\neg u$

$\neg x \vee z$

$x \vee u$



Demo Teaser

Demo Will Show a Higher Order Example

Let's set it up now...

Closure Operators

Source: Chapter 1 of Someone's Dissertation.

Closure Operators

Source: Chapter 1 of Someone's Dissertation.

Definition 1.1 *Let M be an arbitrary set. A function $\mathfrak{cl} : \mathcal{P}(M) \to \mathcal{P}(M)$ will be called* CLOSURE OPERATOR *on M if it is*

1. EXTENSIVE, *i.e., $A \subseteq \mathfrak{cl}(A)$ for all $A \subseteq M$,*
2. MONOTONE, *i.e., $A \subseteq B \Rightarrow \mathfrak{cl}(A) \subseteq \mathfrak{cl}(B)$ for all $A, B \subseteq M$, and*
3. IDEMPOTENT, *i.e., $\mathfrak{cl}(\mathfrak{cl}(A)) = \mathfrak{cl}(A)$ for all $A \subseteq M$.*

A set $A \subseteq M$ will be called CLOSED *(or $\mathfrak{cl}$-*CLOSED *in case of ambiguity), if $\mathfrak{cl}(A) = A$.*
The set of all closed sets $\{A \mid A = \mathfrak{cl}(A) \subseteq M\}$ will be called CLOSURE SYSTEM.

◇

It is easy to show that for an arbitrary closure system $\mathcal{S}$, the corresponding closure operator $\mathfrak{cl}$ can be reconstructed by

$$\mathfrak{cl}(A) = \bigcap_{B \in \mathcal{S}, A \subseteq B} B.$$

Setting Up The Problem

Definition 1.1 *Let M be an arbitrary set. A function* $\mathfrak{cl} : \mathcal{P}(M) \to \mathcal{P}(M)$ *will be called* CLOSURE OPERATOR *on M if it is*

```
sort M

const cl:(M B) M B
```

The type `(M B)` is the type of functions from M to Booleans - characteristic functions of sets in $\mathcal{P}(M)$.

Setting Up The Problem

Definition 1.1 *Let M be an arbitrary set. A function $\mathfrak{cl} : \mathcal{P}(M) \to \mathcal{P}(M)$ will be called* CLOSURE OPERATOR *on M if it is*

```
sort M
```

```
const cl:(M B) M B
```

The type `(M B)` is the type of functions from M to Booleans - characteristic functions of sets in $\mathcal{P}(M)$.

```
var x:M

var A B:M B

term subseteq=(\A B.!x.A x -> B x)

infix subseteq 40 40
```

Setting Up The Problem

be called CLOSURE OPERATOR *on* M *if it is*

1. EXTENSIVE, *i.e.*, $A \subseteq \mathfrak{cl}(A)$ *for all* $A \subseteq M$,

2. MONOTONE, *i.e.*, $A \subseteq B \Rightarrow \mathfrak{cl}(A) \subseteq \mathfrak{cl}(B)$ *for all* $A, B \subseteq M$, *and*

3. IDEMPOTENT, *i.e.*, $\mathfrak{cl}(\mathfrak{cl}(A)) = \mathfrak{cl}(A)$ *for all* $A \subseteq M$.

Setting Up The Problem

be called CLOSURE OPERATOR *on M if it is*

1. EXTENSIVE, *i.e.*, $A \subseteq \mathfrak{cl}(A)$ *for all* $A \subseteq M$,

2. MONOTONE, *i.e.*, $A \subseteq B \Rightarrow \mathfrak{cl}(A) \subseteq \mathfrak{cl}(B)$ *for all* $A, B \subseteq M$, *and*

3. IDEMPOTENT, *i.e.*, $\mathfrak{cl}(\mathfrak{cl}(A)) = \mathfrak{cl}(A)$ *for all* $A \subseteq M$.

```
axiom extensive:!A.A subseteq (cl A)
```

Setting Up The Problem

be called CLOSURE OPERATOR *on* M *if it is*

1. EXTENSIVE, *i.e.*, $A \subseteq \mathfrak{cl}(A)$ *for all* $A \subseteq M$,
2. MONOTONE, *i.e.*, $A \subseteq B \Rightarrow \mathfrak{cl}(A) \subseteq \mathfrak{cl}(B)$ *for all* $A, B \subseteq M$, *and*
3. IDEMPOTENT, *i.e.*, $\mathfrak{cl}(\mathfrak{cl}(A)) = \mathfrak{cl}(A)$ *for all* $A \subseteq M$.

```
axiom extensive:!A.A subseteq (cl A)
axiom monotone:!A B.A subseteq B
                   -> (cl A) subseteq (cl B)
```

Setting Up The Problem

be called CLOSURE OPERATOR *on* M *if it is*

1. EXTENSIVE, *i.e.*, $A \subseteq \mathfrak{cl}(A)$ *for all* $A \subseteq M$,

2. MONOTONE, *i.e.*, $A \subseteq B \Rightarrow \mathfrak{cl}(A) \subseteq \mathfrak{cl}(B)$ *for all* $A, B \subseteq M$, *and*

3. IDEMPOTENT, *i.e.*, $\mathfrak{cl}(\mathfrak{cl}(A)) = \mathfrak{cl}(A)$ *for all* $A \subseteq M$.

```
axiom extensive:!A.A subseteq (cl A)
axiom monotone:!A B.A subseteq B
                    -> (cl A) subseteq (cl B)
axiom idempotent:!A.(cl (cl A))=(cl A)
```

Setting Up The Problem

A set $A \subseteq M$ will be called CLOSED *(or* $\mathfrak{cl}$-CLOSED *in case of ambiguity), if* $\mathfrak{cl}(A) = A$.

```
term closed=(\A.A = cl A)
```

Setting Up The Problem

$$\mathfrak{cl}(A) = \bigcap_{B \in \mathcal{S}, A \subseteq B} B.$$

Setting Up The Problem

$$\mathfrak{cl}(A) = \bigcap_{B \in \mathcal{S}, A \subseteq B} B.$$

```
var D:(M B) B
term setintersect=(\D x.!B.D B -> B x)
```

Setting Up The Problem

$$\mathbf{cl}(A) = \bigcap_{B \in \mathcal{S}, A \subseteq B} B.$$

```
var D:(M B) B

term setintersect=(\D x.!B.D B -> B x)

claim !A.(cl A) =
  (setintersect (\B.closed B & A subseteq B))
```

This is what I will prove in the demo...

Source for Closure Operators Example

... ?

Source for Closure Operators Example

RELATIONAL EXPLORATION

COMBINING DESCRIPTION LOGICS AND FORMAL CONCEPT ANALYSIS FOR KNOWLEDGE SPECIFICATION

Sebastian Rudolph
Institute for Algebra
Faculty of Mathematics and Natural Sciences
TU Dresden

Conclusion

Try it Online:

`http://ps.uni-sb.de/jitpro/`

See You At The Demo