Introduction to Logic
Fitch Proofs

Michael Genesereth
Computer Science Department
Stanford University

Propositional Logic

$$\{m \Rightarrow p \lor q, p \Rightarrow q\} \models m \Rightarrow q?$$

m	p	q	$m \Rightarrow p \lor q$	$p \Rightarrow q$	$m \Rightarrow q$
1	1	1	1	1	1
1	1	0	1	0	0
1	0	1	1	1	1
1	0	0	0	1	0
0	1	1	1	1	1
0	1	0	1	0	1
0	0	1	1	1	1
0	0	0	1	1	1

Relational Logic

$$\{p(a) \lor p(b), \forall x.(p(x) \Rightarrow q(x))\} \models \exists x.q(x)?$$

$p(a)$	$p(b)$	$q(a)$	$q(b)$	$p(a) \lor p(b)$	$\forall x.(p(x) \Rightarrow q(x))$	$\exists x.q(x)$
1	1	1	1	1	1	1
1	1	1	0	1	0	1
1	1	0	1	1	0	1
1	1	0	0	1	0	0
1	0	1	1	1	1	1
1	0	1	0	1	1	1
1	0	0	1	1	0	1
1	0	0	0	1	0	0
0	1	1	1	1	1	1
0	1	1	0	1	0	1
0	1	0	1	1	1	1
0	1	0	0	1	0	0
0	0	1	1	0	1	1
0	0	1	0	0	1	1
0	0	0	1	0	1	1
0	0	0	0	0	1	0

Functional Logic

Object constants: *n*
Unary relation constants: *k*
Factoids in Herbrand Base: *infinite*
Interpretations: *infinite*

Functional Logic Proofs

Logical Rules of Inference

Negation Introduction
Negation Elimination

And Introduction
And Elimination

Or Introduction
Or Elimination

Assumption
Implication Elimination

Biconditional Introduction
Biconditional Elimination

Rules of Inference for Quantifiers

Universal Introduction
Universal Elimination

Existential Introduction
Existential Elimination

Domain Closure
Induction (Linear, Tree, Structural) *New!!*

Structured Proof (tl;dr)

A *structured proof* of a conclusion from a set of premises is a sequence of (possibly nested) sentences terminating in an occurrence of the conclusion at the top level of the proof. Each step in the proof must be either (1) a premise (at the top level), (2) an assumption, or (3) the result of applying an ordinary or structured rule of inference to earlier items in the sequence (subject to the constraints given above).

Example - Checking Cells

Interactive Logic Grid

When clicked, a blank cell goes to a check, a checked cell goes to an ex, and an exed cell goes to a blank. Show that if a cell is blank, it will be blank after three clicks.

	1	2	3	4
1	✓	✗	✓	✓
2	✗		✗	
3	✓	✗		✓
4			✓	✗

Problem

When clicked, a blank cell goes to a check, a checked cell goes to an ex, and an exed cell goes to a blank. Show that if a cell is blank, it will be blank after three clicks.

Givens:

$\forall x{:}(blank(x) \Rightarrow check(c(x)))$

$\forall x{:}(check(x) \Rightarrow ex(c(x)))$

$\forall x{:}(ex(x) \Rightarrow blank(c(x)))$

Goal:

$\forall x{:}(blank(x) \Rightarrow blank(c(c(c(x)))))$

Proof

1.	$\forall x{:}(blank(x) \Rightarrow check(c(x)))$	Premise
2.	$\forall x{:}(check(x) \Rightarrow ex(c(x)))$	Premise
3.	$\forall x{:}(ex(x) \Rightarrow blank(c(x)))$	Premise
4.	$\vert\ blank([d])$	Assumption
5.	$\vert\ blank([d]) \Rightarrow check(c([d]))$	UE: 1
6.	$\vert\ check(c([d]))$	IE: 5, 4
7.	$\vert\ check(c([d])) \Rightarrow ex(c(c([d])))$	UE: 2
8.	$\vert\ ex(c(c([d])))$	IE: 7, 6
9.	$\vert\ ex(c(c([d]))) \Rightarrow blank(c(c(c([d]))))$	UE: 3
10.	$\vert\ blank(c(c(c([d]))))$	IE: 9, 8
12.	$blank([d]) \Rightarrow blank(c(c(c([d]))))$	II: 4,10
13.	$\forall x.(blank(x) \Rightarrow blank(c(c(c(x)))))$	UI: 12

Example - Spin

Problem

On transition, a particle goes from spin zero to spin positive or spin negative and then goes to spin zero on second transition. Show that if it has spin zero, it will have spin zero after two transitions.

Givens:

$$\forall x{:}(zero(x) \Rightarrow pos(f(x)) \mid neg(f(x)))$$

$$\forall x{:}(pos(x) \Rightarrow zero(f(x)))$$

$$\forall x{:}(neg(x) \Rightarrow zero(f(x)))$$

Goal:

$$\forall x{:}(zero(x) \Rightarrow zero(f(f(x))))$$

Proof

1.	$\forall x.(zero(x) \Rightarrow pos(f(x)) \vee neg(f(x)))$	Premise
2.	$\forall x.(pos(x) \Rightarrow zero(f(x)))$	Premise
3.	$\forall x.(neg(x) \Rightarrow zero(f(x)))$	Premise

Proof

1.	$\forall x.(zero(x) \Rightarrow pos(f(x)) \vee neg(f(x)))$	Premise
2.	$\forall x.(pos(x) \Rightarrow zero(f(x)))$	Premise
3.	$\forall x.(neg(x) \Rightarrow zero(f(x)))$	Premise
4.	$\mid zero([c])$	Assumption

Proof

1.	$\forall x.(zero(x) \Rightarrow pos(f(x)) \vee neg(f(x)))$	Premise
2.	$\forall x.(pos(x) \Rightarrow zero(f(x)))$	Premise
3.	$\forall x.(neg(x) \Rightarrow zero(f(x)))$	Premise
4.	$zero([c])$	Assumption
5.	$zero([c]) \Rightarrow pos(f([c])) \vee neg(f([c]))$	UE: 1

Proof

1.	$\forall x.(zero(x) \Rightarrow pos(f(x)) \vee neg(f(x)))$	Premise
2.	$\forall x.(pos(x) \Rightarrow zero(f(x)))$	Premise
3.	$\forall x.(neg(x) \Rightarrow zero(f(x)))$	Premise
4.	$\quad zero([c])$	Assumption
5.	$\quad zero([c]) \Rightarrow pos(f([c])) \vee neg(f([c]))$	UE: 1
6.	$\quad pos(f([c])) \vee neg(f([c]))$	IE: 5, 4

Proof

1.	$\forall x.(zero(x) \Rightarrow pos(f(x)) \vee neg(f(x)))$	Premise
2.	$\forall x.(pos(x) \Rightarrow zero(f(x)))$	Premise
3.	$\forall x.(neg(x) \Rightarrow zero(f(x)))$	Premise
4.	$zero([c])$	Assumption
5.	$zero([c]) \Rightarrow pos(f([c])) \vee neg(f([c]))$	UE: 1
6.	$pos(f([c])) \vee neg(f([c]))$	IE: 5, 4
7.	$pos(f([c])) \Rightarrow zero(f(f([c])))$	UE: 2
8.	$neg(f([c])) \Rightarrow zero(f(f([c])))$	UE: 3

Proof

1.	$\forall x.(zero(x) \Rightarrow pos(f(x)) \vee neg(f(x)))$	Premise
2.	$\forall x.(pos(x) \Rightarrow zero(f(x)))$	Premise
3.	$\forall x.(neg(x) \Rightarrow zero(f(x)))$	Premise
4.	$zero([c])$	Assumption
5.	$zero([c]) \Rightarrow pos(f([c])) \vee neg(f([c]))$	UE: 1
6.	$pos(f([c])) \vee neg(f([c]))$	IE: 5, 4
7.	$pos(f([c])) \Rightarrow zero(f(f([c])))$	UE: 2
8.	$neg(f([c])) \Rightarrow zero(f(f([c])))$	UE: 3
9.	$zero(f(f([c])))$	OE: 6, 7, 8

Proof

1. $\forall x.(zero(x) \Rightarrow pos(f(x)) \vee neg(f(x)))$ — Premise
2. $\forall x.(pos(x) \Rightarrow zero(f(x)))$ — Premise
3. $\forall x.(neg(x) \Rightarrow zero(f(x)))$ — Premise
4. $\quad zero([c])$ — Assumption
5. $\quad zero([c]) \Rightarrow pos(f([c])) \vee neg(f([c]))$ — UE: 1
6. $\quad pos(f([c])) \vee neg(f([c]))$ — IE: 5, 4
7. $\quad pos(f([c])) \Rightarrow zero(f(f([c])))$ — UE: 2
8. $\quad neg(f([c])) \Rightarrow zero(f(f([c])))$ — UE: 3
9. $\quad zero(f(f([c])))$ — OE: 6, 7, 8
10. $zero([c]) \Rightarrow zero(f(f([c])))$ — II: 4, 9

Proof

1.	$\forall x.(zero(x) \Rightarrow pos(f(x)) \lor neg(f(x)))$	Premise
2.	$\forall x.(pos(x) \Rightarrow zero(f(x)))$	Premise
3.	$\forall x.(neg(x) \Rightarrow zero(f(x)))$	Premise
4.	$\quad zero([c])$	Assumption
5.	$\quad zero([c]) \Rightarrow pos(f([c])) \lor neg(f([c]))$	UE: 1
6.	$\quad pos(f([c])) \lor neg(f([c]))$	IE: 5, 4
7.	$\quad pos(f([c])) \Rightarrow zero(f(f([c])))$	UE: 2
8.	$\quad neg(f([c])) \Rightarrow zero(f(f([c])))$	UE: 3
9.	$\quad zero(f(f([c])))$	OE: 6, 7, 8
10.	$zero([c]) \Rightarrow zero(f(f([c])))$	II: 4, 9
11.	$\forall x.(zero(x) \Rightarrow zero(f(f(x))))$	UI: 10

Analysis

Logical Entailment and Provability

A set of premises Δ *logically entails* a conclusion φ ($\Delta \models \varphi$) if and only if every truth assignment that satisfies Δ also satisfies φ.

If there exists a proof of a sentence ϕ from a set Δ of premises using the rules of inference in R, we say that ϕ is *provable* from Δ using R (written $\Delta \vdash_R \phi$).

Soundness and Completeness

A proof system is *sound* if and only if every provable conclusion is logically entailed.

$$\text{If } \Delta \vdash \phi\text{, then } \Delta \vDash \phi.$$

A proof system is *complete* if and only if every logical conclusion is provable.

$$\text{If } \Delta \vDash \phi\text{, then } \Delta \vdash \phi.$$

Fitch Proof System

Theorem: Fitch is sound and complete for **Relational Logic**.

$$\Delta \models \varphi \text{ if and only if } \Delta \vdash_{\text{Fitch}} \phi.$$

Theorem: Fitch is *sound* for **Functional Logic** but it is *not complete*.

$$\text{if } \Delta \vdash_{\text{Fitch}} \phi, \text{ then } \Delta \models \varphi.$$

Bad News

Theorem: Satisfiability / logical entailment for FL are not decidable.

Theorem: There is *no* sound and complete proof procedure for Functional Logic.

Diophantine Equations

Diophantine equation:

$$3x^2 = 1$$

Diophantine equation in Peano Arithmetic:

$$\exists x.\exists y.(\mathit{times}(x,x,y) \wedge \mathit{times}(s(s(s(0))),y,s(0)))$$

Solvability Question:

Axioms of Arithmetic $\cup$
$\{\exists x.\exists y.(\mathit{times}(x,x,y) \wedge \mathit{times}(s(s(s(0))),y,s(0)))\}$

It is known that the question of solvability of Diophantine equations is *not* decidable.

Two Types of Completeness

A *proof procedure* is complete if and only if everything that is logically entailed is provable.

A *set of sentences* Δ is complete if and only if for every sentence ϕ, either $\Delta \models \phi$ or $\Delta \models \neg\phi$.

Informal Proof

Given a finite set of sentences and a complete proof procedure, logical entailment is semidecidable.

If the set of sentence is complete, then logical entailment is decidable. (We try to prove the sentence and its negation, and one of the two will terminate if complete.)

Our axiomatization of Peano Arithmetic in Functional Logic is finite and complete. **Assume** we had a sound and complete proof procedure. Then we would be able to decide the satisfiability of any Diophantine equation. **Contradiction**. Upshot: there is no sound and complete proof procedure for Functional Logic.

Comparison

Even though there is no complete proof procedure, Functional Logic is ***not*** inferentially weaker than Relational Logic and First-Order Logic. In fact, it is stronger than Relational Logic or First-Order Logic.

The problem is that there are *more* things that are true. We cannot prove them *all*, but we can prove everything we could prove in First Order Logic; and, by building in induction, we can prove *more* things.

Compactness

A logic is *compact* if and only if every unsatisfiable set of sentences has a finite subset that is unsatisfiable.

Theorem: Propositional Logic is compact.
Theorem: Relational Logic is compact.
Theorem: Functional Logic is *not* compact.

$$\{p(0), p(s(0)), p(s(s(0))), \ldots, \exists x.\neg p(x)\}$$

Significance: Some conclusions in Functional Logic have only infinite proofs.