Lecture 20

Pete Manolios
Northeastern

Question 2

```
(defdata lon (listof nat))
(defdata nlon (cons nat lon))
(defunc f (l)
  :input-contract (nlonp l)
  :output-contract (natp (f l))
   (cond ((endp (cdr l)) (car l))
         ((equal (car l) 0) (f (cdr l)))
         ((equal (car l) (+ 1 (second l)))
          (f (cdr l)))
         (t (f (cons (car l) (cons (- (car l) 1) (cdr l)))))))
```

No one came up with a measure that works!

```
(defunc m (l)
  :input-contract (nlonp l)
  :output-contract (natp (m l))
  (cond ((endp (cdr l)) 0)
        ((equal (car l) 0) (+ 1 (m (cdr l))))
        ((equal (car l) (+ 1 (second l)))
         (+ 1 (m (cdr l))))
        (t (+ 1 (* 2 (car l)) (m (cdr l))))))
```

Almost all proposed measures had simple counterexamples, so test your measure functions!

A useful pattern: use the same cond structure as the function you want to admit

Question 3

```
(A (A (R (A x y)) z) w)
= { R1 }
(A (A (A (R y) (R x)) z) w)
= { R2 }
(A (A (A (R x) (R y)) z) w)
= { R3 }
(A (A (R x) (A (R y) z)) w)
= { R2 }
(A (A (R x) (A z (R y))) w)
= { R3 }
(A (R x) (A (A z (R y)) w))
= { R3 }
(A (R x) (A z (A (R y) w)))
= { R2 }
(A (R x) (A z (A w (R y))))
```

```
R1. (R (A x y)) = (A (R y) (R x))
R2. (A y x) = (A x y)
R3. (A (A x y) z) = (A x (A y z))
```

Rewriting is the most important part of ACL2, so remember:

1. *Left to right (everyone got that)*
2. *Inside-out*
3. *Reverse chronological*

*Plus special handling of permutative rules, type reasoning, linear arithmetic, tau, conditional rewriting, forward chaining, …
(most of which I didn't test)*

Question 4

BDD question: Almost everyone got this one right!

Question 5

DP question: Four of you struggled, but mostly easy.
Hopefully easy now after implementing DP.

Question 6

Proof question: Only 1 person got a perfect score.
All but two people got more than 1/2 credit.

Surprisingly, most of you came up with the wrong lemma!
Simple counterexamples exist, so write tests for lemmas.

The lemma generation project by Ben is relevant

Gödel's Completeness Theorem

- $\Phi \vdash \phi$ iff $\Phi \models \phi$
- What does this mean for group theory?
- What about new proof techniques?
- Once we show the equivalence between $\vdash \phi$ and $\models$, we can transfer properties of one to the other
  - Compactness theorem:
    (a) $\Phi \models \phi$ iff there is a finite $\Phi_0 \subseteq \Phi$ such that $\Phi_0 \models \phi$
    (b) Sat Φ iff for all finite $\Phi_0 \subseteq \Phi$, Sat Φ_0
- From the proof, we get the Löwenheim-Skolem theorem: Every satisfiable and at most countable set of formulas is satisfiable over a domain which is at most countable

Consequences of Completeness

Theorem *Every satisfiable set of formulas* $\Phi \subseteq L^S$ *is satisfiable over a domain of cardinality* $\leq |L^S|$.

Theorem *If* $\Phi \subseteq L_0^S$ *has arbitrarily large finite models (for every* $i \in \omega$, Φ *has a model whose domain has more than* i *elements), then it has an infinite model.*

Proof Let

$$\psi_2 = \exists x \exists y \; x \not\equiv y$$

$$\psi_3 = \exists x \exists y \exists z \; (x \not\equiv y) \wedge (x \not\equiv z) \wedge (y \not\equiv z)$$

$$\ldots$$

Consider $\Phi \cup \{\psi_i : i > 2\}$. Note that every finite subset has a model. By compactness, the set has a model. $\square$

Theories

T *is a theory iff* $T \subseteq L_0^S$ *and for all* $\varphi \in L_0^S$, $T \models \varphi \Rightarrow \varphi \in T$.

$\Phi^{\models} = \{\varphi : \Phi \models \varphi\}$.

T *is a theory iff* $T = T^{\models}$

T *is* complete *iff* $\langle \forall \varphi :: \varphi \in T \vee \neg\varphi \in T \rangle$.

For a set of structures $\mathbf{K}$, $\mathrm{Th}\,\mathbf{K} = \{\varphi \in L_0^S : \forall\, \mathbf{U} \in \mathbf{K}, \mathbf{U} \models \varphi\}$

Axiomatizable Theories

Definition *Theory T is* axiomatizable *iff there exists a* decidable *set Φ of sentences s.t. $T = \Phi^{\models}$.*

Definition *Theory T is* finitely axiomatizable *iff $T = \Phi^{\models}$, where Φ is a set of sentences s.t. $|\Phi| < \omega$.*

Note that if Theory T is finitely axiomatizable $T = \{\varphi\}^{\models}$ (where φ is a conjunction of finitely many sentences).

Lemma *(a) If T is finitely axiomatizable, it is axiomatizable. (b) If T is axiomatizable, it is r.e. (c) If T is axiomatizable and complete, it is recursive.*

Non-Standard Models

- Let $N_s = \langle \omega, s, 0 \rangle$, where s is the successor function. N_s satisfies:
  - (the successor of any number differs from that number) $\langle \forall x\ x \neq s(x) \rangle$
  - (s is injective) $\langle \forall x,y\ x \neq y \ \Rightarrow\ s(x) \neq s(y) \rangle$
  - (every non-0 number has a predecessor) $\langle \forall x\ x \neq 0 \ \Rightarrow\ \langle \exists y\ y = s(x) \rangle \rangle$
- Let $\Psi = \text{Th}\ N_s \cup \{x \neq 0, x \neq s(0), \ldots, x \neq s^n(0), \ldots\}$
- Every finite subset of Ψ has a model, so Ψ has a model (compactness)
- By Lowenheim-Skolem, let $\mathfrak{U}$ be a countable model of Ψ
  - $\mathfrak{U}$ includes $0, s(0), \ldots, s^n(0), \ldots$, and a, a non-standard number
  - a has a successor, predecessor, and they have successors, predecessors
  - so a is part of a $\mathbb{Z}$-chain
  - hence, there is a countable model, $\mathfrak{U}$, which is *not* isomorphic to N_s
- While there is a complete axiomatization for Th N_s, once the logic is powerful enough (add +, *, <), completeness goes out the window

$0, s(0), \ldots, s^n(0), \ldots,$ $\quad \ldots, p^n(a), \ldots, p(a), a, s(a), \ldots, s^n(a), \ldots$ $\quad$ $\mathbb{Z}$-chain

$p(a)$ is the predecessor of a $\quad$ (isomophic to $\mathbb{Z}$)

Gödel's 1st Incompleteness Theorem

- A set is *recursive* iff $\in$ can be decided by a Turing machine
- Assuming Con(ZF), the set $\{\phi : ZF \vdash \phi\}$ is not recursive
- More generally, for any consistent extension C of ZF:
  - $\{\phi : C \vdash \phi\}$ is not recursive
  - Intuitively clear: embed Turing machines in set theory
  - Encode halting problem! as a formula in set theory
- Theorem: If C is a recursive consistent extension of ZF, then it is incomplete, i.e., there is a formula ϕ such that $C \nvdash \phi$ and $C \nvdash \neg\phi$
- Proof Outline: If not, then for every ϕ, either $C \vdash \phi$ or $C \vdash \neg\phi$. We can now decide $C \vdash \phi$: enumerate all proofs of C. Stop when a proof for ϕ or $\neg\phi$ is found

FOL Observations

- In ZF, the axiom of choice is neither provable nor refutable
- In ZFC, the continuum hypothesis is neither provable nor refutable
- By Gödel's first incompleteness theorem, no matter how we extend ZFC, there will always be sentences which are neither provable nor refutable
- There are non-standard models of $\mathbb{N}$, $\mathbb{R}$ (un/countable)
- Since any reasonable proof theory has to be decidable, and TMs can be formalized in FOL (set theory), any logic can be reduced to FOL
- Building reliable computing systems requires having programs that can reason about other programs and this means we have to really understand what a proof is so that we can program a computer to do it

Presentation/Project Schedule

- 11/27
  - Ben B (40 min)
  - Dustin (40 min)
  - Alex (20 min)
- 11/30
  - Ankit (40 min)
  - Taylor (20 min)
  - Nathaniel (20 min)
  - Daniel (20 min)
- 12/4
  - Michael (20 min)
  - Drew (40 min)
  - Ben Q (40 min)

Meet with me to review slides at least 3 days before your presentations

Exam 2:

Distribute 11/30 after class

Due 12/1 by 3PM (email)