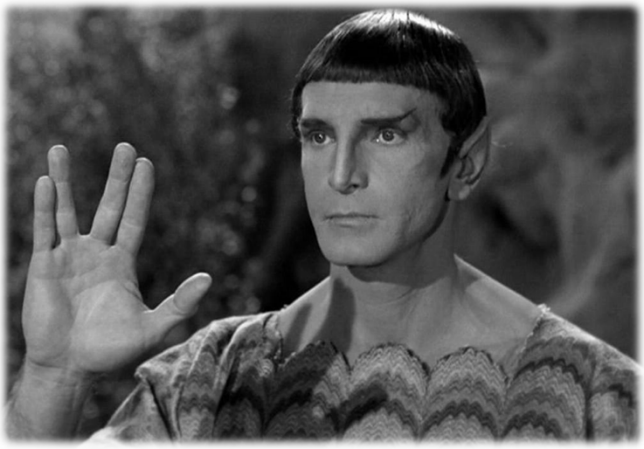
Chapter 1: Preliminaries

Presented by David Brown

Chapter overview

- Sections:
  - Logic formulas
  - Semantics of Formulas
  - Models and Logical Consequence
  - Logical Inference
  - Substitutions
- Big ideas (first two sections):
  - Logic, Inference, and Formal Logic
  - Syntax
  - Semantics

The discussion of Chapter 1 will be split into two sessions; the first will cover 1.1 and 1.2; the second will cover 1.3, 1.4, and 1.5.



Logic

Surak, a legendary logician in the fictional Star Trek universe

http://memory-alpha.org/wiki/File:Surak_TOS.jpg. Copyright Paramount/CBS. Actually, Vulcan “logic” isn’t going to help us here at all.

What is Logic?

- A dozen definitions collected by Wikipedia make repeated reference to:
  - "true," "false," "truth"
  - "reason," "reasoning"
  - "thought," "operations of the mind," "understanding"
  - "science"
  - "art" (?)
  - "laws," "rules"
- Informally, "Logic is the study of what counts as a good reason, for what, and why."

Priest, Graham. Logic: A very Short Introduction. Oxford, 2000.

Reasoning by inference

- Premises + Inference = conclusion
- Example:

1. All men are mortal. (*premise*)
2. Socrates is a man. (*premise*)
3. *(apply appropriate rules of inference)*
4. Therefore, Socrates is mortal. (*conclusion*)

- More on inference in section 1.4…

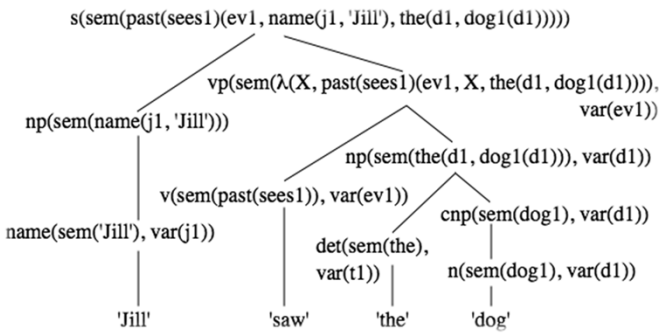
Why we need a formal system

- Another example:

1. Nothing is better than lifelong happiness.
2. A cheese sandwich is better than nothing.
3. Therefore, a cheese sandwich is better than lifelong happiness.

- Natural languages can be imprecise
- A formal language has a defined, simple syntax and reasonably straightforward semantic rules.

Example from: Gowers, ed. Princeton Companion to Mathematics, page 14. “Simple” (syntax) relative to natural languages, that is.



Syntax

A parse tree for a Prolog program in natural language processing (Chapter 9 of Allen, J.: Natural Language Understanding, 2nd ed., Benjamin Cummings, 1995.)

http://www.cse.unsw.edu.au/~billw/cs9414/notes/nlp/seminterp/seminterp-2009.html

Syntax

- In natural language, syntax provides the rules by which words can be formed into understandable (grammatical) sentences.
- In a logic language, syntax provides the rules by which symbols can be arranged into well-formed formulas (*wff*s).
- To define the syntax, we start with *symbols*, collect them in an *alphabet*, identify the *terms* we can form from the alphabet, and finally construct *formulas*.

About symbols

- A symbol is a signifier: a token that represents something else
- Some symbols are created in the structure of a specific program; these denote things and relations in the world
- Other symbols already exist in the logic system itself; these access the computational mechanism

The symbols mentioned here seem to correspond to the Alpha and the Omega sets of a propositional calculus.

Symbols that denote things

- Variables:
  - Refer to some unspecified thing: *X*
- Constants:
  - denote things in the world: *tom*
- Functor:
  - denote composite things: *family*/3, *child*/2
    family(*bill*,*mary*(*child*(*tom*,*child*(*alice*,*none*))))
- Predicates:
  - denote relations in the world: *child_of*/2
    child_of (*tom*,*mary*)

The family/3 and child/2 functors are particularly awkward first examples; they shows, though, how you could use functors to build a list of arbitrary length.

Symbols to operate the system

- Logical connectives (operators)
  - ∧ (conjunction), ∨ (disjunction), ¬ (negation)
  - ↔ (logical equivalence), ⊃ (implication)
- Quantifiers
  - ∀: universal quantifier “for all…”
  - ∃: existential quantifier “there exists some…”
- Auxiliary symbols (Prolog)
  - used to facilitate notation; examples include…
  - parentheses to group objects of a predicate
  - comma to separate terms
  - whitespace

Alphabet

- Definition 1.0 (Alphabet):
  The alphabet ($\mathcal{A}$) consists of…
  - all the symbols for constants, variables, functors, and predicates
    - In Prolog, the alphanumeric label (*tom, family*) is an "identifier"
    - Functors and predicates also have an "arity" – the number of arguments they take. This is notated as *f*/*n* where *f* is the identifier and *n* is the arity.
    - Multiple distinct objects can have the same identifier when their arity differs
  - all the symbols for operating the computation (connectives, quantifiers, auxiliaries)

This definition of an alphabet isn't actually called out by number in the book. While Nilsson includes the connectives/quantifiers/aux in the alphabet, I can't find where that is actually used.

Terms

- Definition 1.1: The set of terms $\mathcal{T}$ over a given alphabet $\mathcal{A}$ is the smallest set such that:
  - any constant in $\mathcal{A}$ is in $\mathcal{T}$;
  - any variable in $\mathcal{A}$ is in $\mathcal{T}$;
  - if f/n is a functor in $\mathcal{A}$ and $t_1 \ldots t_n \in \mathcal{T}$ then $f(t_1 \ldots t_n) \in \mathcal{T}$
- (Predicates are not terms.)

Formulas

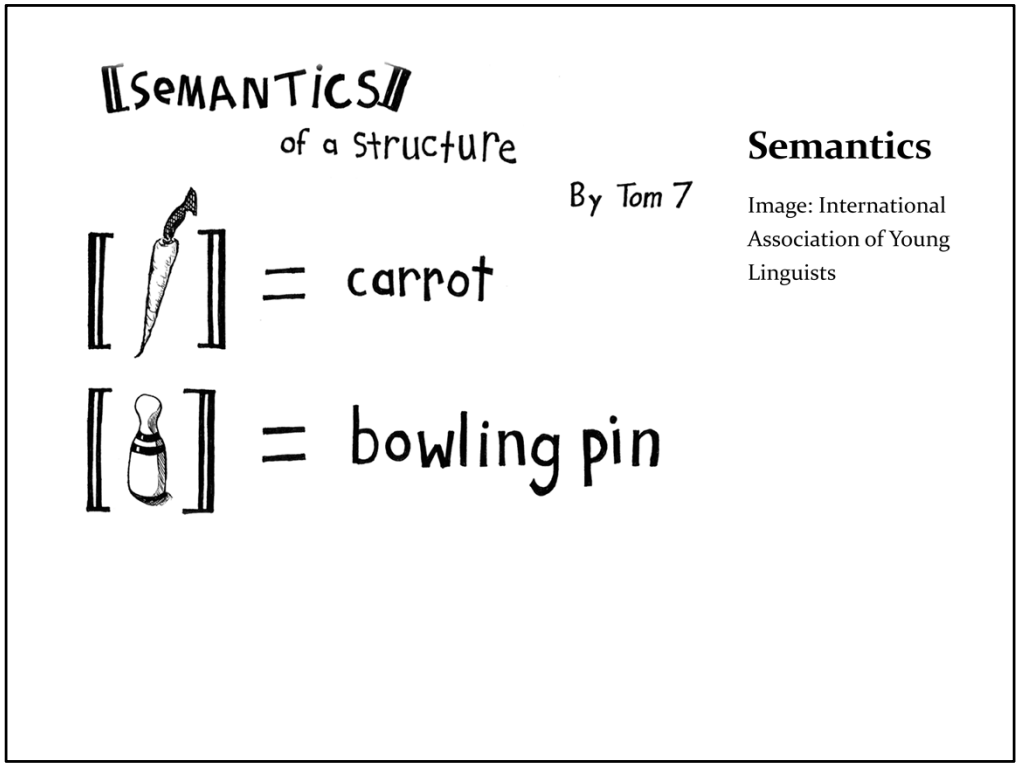
- Definition 1.2: The set $\mathcal{F}$ of wff is the smallest set such that:
  - if p/n is a predicate in $\mathcal{A}$ and $t_1 \ldots t_n \in \mathcal{T}$ then $p(t_1 \ldots t_n) \in \mathcal{F}$
  - if F and $G \in \mathcal{F}$ then so are $(\neg F)$, $(F \wedge G)$, $(F \vee G)$, $(F \supset G)$, and $(F \leftrightarrow G)$;
  - if $F \in \mathcal{F}$ and X is a variable in $\mathcal{A}$ then $(\forall XF)$ and $(\exists XF)$ $\in \mathcal{F}$

1) grab the predicates; 2) grab the basic operators; 3) include quantifiersActually, not-F is in the set of wff even if G isn't.

World-Logic Counterpart Review

The "real world"	Predicate logic system
• universe (world)	• structure
• things (individuals)	• constants
• relations	• predicates
• declarative sentences	• formulas
• words	• symbols
• lexicon (words in the language)	• alphabet (collection of symbols)

THIS SLIDE SHOULD PROBABLY COME LATER AS A SUMMARY



http://www.iayl.org/site/Scientific_Board/Semantics/Semantics_200704187.html

Semantics

- Syntax tells us if a formula is well-formed; semantics tells us the formula's meaning.
- The meaning of a logic formula – a truth value – is determined relative to an abstraction of the "real" world called a *structure*.
- An *interpretation* links constants (and functors) and predicates to the individuals (and composite entities) and relations of the structure.

The meaning of a logic formula is simply a truth value: either true or false. Definitely a "getting there is half the fun" situation!

Structure

- A structure consists of:
  - a nonempty set of individuals (called the domain)
  - relations defined on the domain
  - functions defined on the domain

Interpretation

- Definition 1.3: An interpretation $\mathfrak{J}$ of an alphabet $\mathcal{A}$ is a non-empty domain $\mathcal{D}$ (sometimes denoted $|\mathfrak{J}|$) and a mapping that associates:
  - each constant $c \in \mathcal{A}$ with an element $c_{\mathfrak{J}} \in \mathcal{D}$;
  - each n-ary functor $f \in \mathcal{A}$ with a function $f_{\mathfrak{J}} \in \mathcal{D}^n \to \mathcal{D}$;
  - each n-ary predicate symbol $p \in \mathcal{A}$ with a relation $p_{\mathfrak{J}} \subseteq \mathcal{D} \times \cdots \times \mathcal{D}$.

In linguistic terms, the interpretation establishes the reference from a word/symbol to the thing itself… or at least to the model of the thing itself. Okay, for the predicate thing, I finally found something I couldn't do in Equation Editor.

Semantics of variables

- Variables are either bound or free
  - a variable X that occurs directly after a quantifier or inside the formula which follows directly after $\forall X$ or $\exists X$ is "bound"
  - a free variable can take on the value of some particular object – more on this in section 1.5, substitution
- A formula with no free variables is *closed*.

You might think of a bound variable as being bound to servitude searching through the set of all possible values

Semantics of variables: Valuation

- Variables are resolved by a mapping function (often identified as φ, "phi") called a *valuation*.
- A valuation maps from variables of the alphabet to elements in the domain of the interpretation.
- The notation $\varphi[X \mapsto t]$ denotes a valuation identical to φ except that $\varphi[X \mapsto t]$ maps X to t.

Extending a valuation with a new mapping is used in the semantics for quantification

Semantics of terms

- Definition 1.4: Let $\mathfrak{I}$ be an interpretation, φ a valuation and t a term. The meaning $\varphi_{\mathfrak{I}}(t)$ of t is an element in $|\mathfrak{I}|$ defined as follows:
  - if t is a constant c then $\varphi_{\mathfrak{I}}(t) \coloneqq c_{\mathfrak{I}}$
  - if t is a variable X then $\varphi_{\mathfrak{I}}(t) \coloneqq \varphi(X)$
  - if t is of the form $f(t_1, \dots, t_n)$, then $\varphi_{\mathfrak{I}}(t) \coloneqq f(\varphi_{\mathfrak{I}}(t_1), \dots, \varphi_{\mathfrak{I}}(t_n))$.
- Interesting things to notice:
  - The truth value of a closed formula (no free variables) depends only on the interpretation; the valuation has no effect.
  - The valuation function for functors and predicates $[f(t_1, \dots, t_n)]$ is called recursively.

It appears that for variables and functors, the valuation function may need to be applied multiple times. For constants, nothing changes; the interpretation is sufficient to provide meaning.

Semantics of wff's

- The meaning of a formula is a truth value determined with respect to an interpretation $\mathfrak{I}$ and a valuation φ.
- Definition 1.6: … is too cumbersome to retype (you can find it on page 9). To paraphrase…
  - an atomic formula $p(t_1, \ldots, t_n)$ is true if its counterpart $\langle \varphi_{\mathfrak{I}}(t_1), \ldots, \varphi_{\mathfrak{I}}(t_n) \rangle$ is in the domain
  - negation and the logical connectives work pretty much just as you'd expect
  - Quantification is a little more interesting: $\mathfrak{I} \vDash_{\varphi} (\forall XF)$ iff $\mathfrak{I} \vDash_{\varphi[X \mapsto t]} F$ for every $t \in |\mathfrak{I}|$ (existential quantification simply changes ∀ to ∃ and "every" to "some")

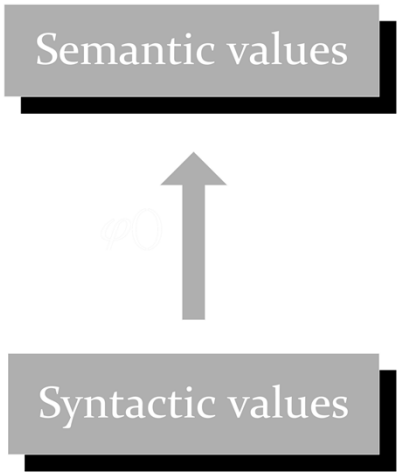
There does not seem to be a definition 1.5 in the book.

Bound vs. Free

- A bound variable has a scope: that part of the formula to the right of its appearance (considering nesting/parentheses)
- A bound variable in a formula gets its value from the quantification process
- Nilsson says "whenever there are free occurrences of variables in a formula its universal closure is considered instead." – I think he means to determine its meaning / truth value in general.
- In Prolog queries, you get an existential closure, returning the first possible solution; rules are still universal closure.

Sections 1.1, 1.2: Logic formulas (syntax), Semantics of Formulas
Big ideas: Logic, Inference, and Formal Logic; Syntax; Semantics

Syntax to Semantics



- For most purposes, the concepts of interpretation ($\mathfrak{I}$), domain ($|\mathfrak{I}|$, or $\mathcal{D}$), and the valuation function (ϕ) can be combined.
- Simply consider $\varphi()$ to be a function that maps from syntax to semantics.

Chapter midpoint overview

- Sections:
  - Logic formulas
  - Semantics of Formulas
  - Models and Logical Consequence
  - Logical Inference
  - Substitutions
- Other Big ideas (along with the headings of the last three sections):
  - Satisfiable, unsatisfiable
  - Logician's toolbox: consequence, equivalence, inference
  - Inconsistent
  - Sound, complete

The discussion of Chapter 1 will be split into two sessions; the first will cover 1.1 and 1.2; the second will cover 1.3, 1.4, and 1.5.

Models

"It may be a model, Captain, but it's highly illogical."

http://www.fieldstonealliance.org/client/tools_you_can_use/07-27-05_Logic_Models.cfm

Model: definition

- **Definition**: An interpretation $\mathfrak{I}$ is said to be a model of [a set of closed formulas] P iff every formula of P is true in $\mathfrak{I}$.
  - Interpretation = domain + mapping
  - Domain = algebraic structure
  - Mapping = from alphabet (syntax) to domain
- Think of model as a noun: a basis for something, like an artist's model is the reference for a portrait.

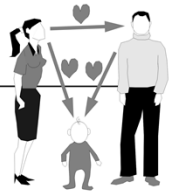
Language is awkward here. I'm used to "model" as a verb: the world exists and I try to model it in functions.

Model: satisfiability

- A set of closed formulas P can have any number of interpretations
- When P is a "proper account of this world" (i.e., the interpretation), the interpretation is a model of P
- There can be any number of models of a *satisfiable* set of formulas.
- If the formulas include some contradiction – e.g., $(F \wedge \neg F)$ – then the set is *unsatisfiable* and can have no models

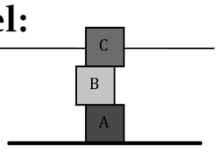
Two models of one set of formulas

Family model:



- $\varphi(tom) \rightarrow$ ⟨Tom⟩
- $\varphi(john) \rightarrow$ ⟨John⟩
- $\varphi(mary) \rightarrow$ ⟨Mary⟩
- $\varphi(loves/2) \rightarrow$ ⟨loves(Lover,Beloved)⟩
- $\varphi(child\_of/2) \rightarrow$ ⟨child_of(Child,Parent)⟩
- $\varphi(mother/1) \rightarrow$ ⟨mother(Mom)⟩

Boxes model:



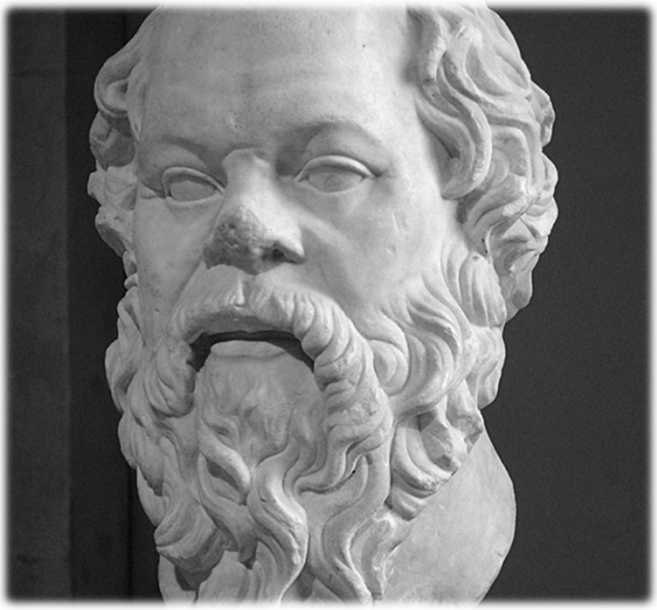
- $\varphi(tom) \rightarrow$ ⟨A⟩
- $\varphi(john) \rightarrow$ ⟨B⟩
- $\varphi(mary) \rightarrow$ ⟨C⟩
- $\varphi(loves/2) \rightarrow$ ⟨is_above(Upper,Lower)⟩
- $\varphi(child\_of/2) \rightarrow$ ⟨is_below(Lower,Upper)⟩
- $\varphi(mother/1) \rightarrow$ ⟨is_on_top(Top)⟩

Formulas:

1) $\forall X \left(\forall Y \left((mother(X) \wedge child\_of(Y,X)) \supset loves(X,Y)\right)\right)$

2) $mother(mary) \wedge child\_of(tom,mary)$

3) $loves(mary,tom)$



Logical Reasoning

A portrait statue of Socrates in the Louvre

Consequence, equivalence, and inference

Logical consequence

- **Definition**: Let P be a set of closed formulas. A closed formula F [$F \notin P$] is called a logical consequence of P (denoted $P \models F$) iff F is true in ***every*** model of P.
- Slight problem here: if $P \models F$ then there exist an infinite number of models of $\{P \cup F\}$; could take a while to check them all! (We'll get some help soon.)
- Unsatisfiability proposition might help a little: $P \models F$ iff $(P \cup \{\neg F\})$ is unsatisfiable
- To show that F is not a logical consequence of P ($P \not\models F$) is easier: show ***any*** model of P which is not a model of F.

Logical equivalence

- **Definition**: Two formulas F and G are said to be logically equivalent (denoted $F \equiv G$) iff F and G have the same truth value for all interpretations $\mathfrak{I}$ and valuations φ.
- Applies to bound and to unbound formulas.
- A number of equivalences – simple rewrite rules – are given, including:
  - Versions of DeMorgan's law for quantification operations
  - a sort of commutative property of universal quantification and disjunction:

    $$\forall X\big(F \vee H(X)\big) \equiv F \vee \forall X\big(H(X)\big)$$

    (when F is closed w.r.t. X)

Logical inference

- Formalized "reasoning principles;" re-write rules that can be used to generate new formulas from given ones
- Must be *sound*:
  - whenever the premises are true, the conclusion obtained must be true in the same world
  - produces only logical consequences of the premises
- Examples (proven from semantic definitions):
  - elimination rule for implication *(modus ponens)*: $\frac{F \quad F \supset G}{G}$
  - elimination rule for universal quantifier: $\frac{\forall X\, F(X)}{F(t)}$
  - introduction rule for conjunction: $\frac{F \quad G}{F \wedge G}$

Logical inference

- A formula F that is obtained by applying inference rules to a set of formulas P is *derivable* from P, denoted: $P \vdash F$ (also, especially in semantics, "P entails F")
- If the inference rules are *sound*, then any derived formula must be a logical consequence of the premises: $P \models F$ whenever $P \vdash F$
- To be *complete,* a set of inference rules must be able to derive all logically consequent formulas from a set of premises: $P \vdash F$ whenever $P \models F$
- "Soundness is an essential requirement; completeness isn't always possible. … Completeness is one of the properties that makes first-order logic nice." (B. Partee)

Barbara H. Partee, Distinguished University Professor Emerita of Linguistics and Philosophy University of Massachusetts Ling 726: Mathematical Linguistics, Lecture 10 Model Theory // V. Borschev and B. Partee, October 23, 2001
people.umass.edu/partee/726_01/lectures/lecture10.pdf

Inconsistent premises

- Nilsson: "A set of premises is said to be *inconsistent* if any formula can be derived from the set. Inconsistency is the proof-theoretic counterpart of unsatisfiability, and when the inference system is both sound and complete the two are frequently used as synonyms." (emphasis added)
- McGrew: "We resolve this [what happens when the set of premises is inconsistent] by specifying the notion of consequence a bit more tightly: the conclusion is a consequence of the premises if and only if it is impossible for all of the premises to be true and the conclusion to be false. When the premises are inconsistent, this requirement is automatically satisfied (since it is impossible for all of the premises to be true). So on the tightened definition, any formula is a consequence of an inconsistent set of premises."

Might want to skip this slide if running short on time.
Timothy J. McGrew (Western Michigan University, Philosophy Department):
http://homepages.wmich.edu/~mcgrew/Logiprob.htm
By the same tightened "notion of consequence" any conclusion from an empty set of premises is also logically true.

Substitutions

Stick figure substitution cipher from Sir Arthur Conan Doyle's Sherlock Holmes' tale of "The Adventure of the Dancing Men."

(Translated: "AM HERE ABE SLANEY")

http://www.zazzle.co.uk/think_substitution_cryptography_dancing_men_tshirt-235708411090292608

A substitution cipher is a far simpler sort of substitution that we're going to be looking at.

Substitutions

- Substitutions map free variables to terms (constants, variables, and functors… but not to predicates).
- **Definition**: A *substitution* is a finite set of pairs of terms $\{X_1/t_1, \ldots, X_n/t_n\}$ where each t_i is a term and each X_i a variable such that:
  - $X_i \neq t_i$ (can't map a variable to itself), and
  - $X_i \neq X_j$ if $i \neq j$ (only one mapping per variable).
- The *empty substitution* is denoted ε.
- **Definition:** The *application* $E\theta$ of a substitution θ ("theta") to a formula E is the term or formula obtained by simultaneously replacing t_i for every free occurrence of X_i in E. $E\theta$ is called an *instance* of E.

$$X\theta := \begin{cases} t \text{ if } X/t \in \theta \\ X \text{ otherwise} \end{cases}$$

A substitution seems to be essentially the same as the state as discussed in CSC 501.
Application of a substitution is like an operation of the state function.
Warning: Nilsson uses the format X/t but I found many writings online that reverse that:
Theorem: For all formulas , I |= |{t1/x1,...,tn/xn} iff I{t1/x1, . . . , tn/xn} |= .

Substitutions

- Composition: Given two substitutions σ ("sigma") and τ ("tau") and some formula F, $F\sigma\tau = (F\sigma)\tau$
- Examples of application:
  - let $\sigma \coloneqq \{X/foo(Y)\}$ and $\tau \coloneqq \{Y/b\}$:
    $$F(X,Y)\sigma\tau = \big(F(foo(Y),Y)\big)\tau = F(foo(b),b)$$
  - contrast with single substitution: let $\theta \coloneqq \{X/foo(Y)\,, Y/b\}$:
    $$F(X,Y)\theta = F(foo(Y),b)$$
- Definition (of a composed substitution):
  let θ $\coloneqq \{X_1/s_1\,, \ldots, X_n/s_n\}$ and $\sigma \coloneqq \{Y_1/t_1\,, \ldots, Y_n/t_n\}$.
  $$\theta\sigma \coloneqq (\{X_1/s_1\sigma\,, \ldots, X_n/s_n\,\sigma\} \mid X_i \neq s_i\sigma) \cup (\{Y_1/t_1\,, \ldots, Y_n/t_n\} \mid Y_i \notin \{X_1 \ldots X_n\})$$
- Informally…
  - Step 1: Apply σ to each /*term* in θ
  - Step 2: Remove any resulting mappings of *X*/*X* (can't map a variable to itself)
  - Step 3: Include also mappings from σ, but only where *Y* is **not** an *X* in θ (only one mapping per variable).

It's easier to look at application of composition first. Note that composition of substitutions is not commutative.

Honestly, Nilsson didn't do too well with composition. The version on page 15 had an error and its replacement in the errata is still hard to understand. I've modified it somewhat (substitution is simultaneous; $foo(Y)$ is not affected by *Y/b*.)

Examples from 2009 lecture notes for CS 141, Intro to AI taught by Amy Greenwald at Brown University.

http://www.cs.brown.edu/courses/cs141/old/2009/lectures/first_order_logic.pdf

Substitutions

- Substitutions do not, cannot, and may not change the meaning (truth value) of a formula (else the system would be unsound)
  - Substitutions cannot be made for bound variables.
  - Substitutions cannot be made that bind variables.

❧ *finis* ❧

END WITH THIS SLIDE!!