Notes on Satisfiability-Based Problem Solving

First Order Logic

David Mitchell
mitchell@cs.sfu.ca

September 27, 2019

This is a preliminary draft. Please do not distribute. Corrections and suggestions welcome.

In this section we present the basics of classical first order logic. The treatment is similar to that of standard mathematical logic texts, but with a focus on properties that are directly relevant to our context.

1 Formulas of First Order Logic

It is essential to distinguish syntax from semantics. The syntax of first order logic defines the set of strings that are formulas. Our primary interest is in using formulas to define classes of structures, and we start by selecting a suitable vocabulary for describing the structures of interest. A vocabulary $\mathcal{L}$ is a tuple of symbols consisting of:

1. A non-empty set of relation (or predicate) symbols, each with an associated arity.
2. A set (possibly empty) of function symbols, each with an associated arity.

We typically use P,Q,R, etc., for relation symbols, and also symbols such as $=$, $\leq$, and words in camel font. We use f, g, etc., for function symbols, and also symbols such as $+$ and $-$. 0-ary function symbols are constant symbols, for which we use c, 0, 1, etc.

Example 1.

1. *The vocabulary of graphs is $\mathcal{L}_G = (E, =)$, where E and $=$ are both binary relation symbols. E is the edge relation.*
2. *The standard vocabulary for arithmetic, $\mathcal{L}_A = (0, s, +, \cdot\ ; =)$, has the constant symbol 0, the unary function symbol s (for "successor"), the binary function symbols $+$ and $\cdot$, and the binary relation symbol $=$.*

The formulas of first order logic for a given vocabulary $\mathcal{L}$ (or $\mathcal{L}$-formulas) are strings over an alphabet consisting of the symbols in $\mathcal{L}$, a countably infinite set $x_1, x_2, \ldots$ of variable symbols, and the symbols $\wedge, \vee, \neg, \forall, \exists, (,)$. We often use x, y, z etc. for variables.

We first define the *terms*, which are expressions used to denote objects in the universe of discourse, and functions over those objects.

Definition 1. The terms for vocabulary $\mathcal{L}$ (or $\mathcal{L}$-terms) are defined inductively by the following rules.

1. Each variable symbol x_i is an $\mathcal{L}$-term;
2. If f is a k-ary function symbol of $\mathcal{L}$ and $t_1, \ldots t_k$ are $\mathcal{L}$-terms, then $ft_1 \ldots t_k$ is an $\mathcal{L}$-term.

Example 2.

1. *$\mathcal{L}_G$ has no function symbols, so the $\mathcal{L}_G$-terms are just the variable symbols.*
2. *If $\mathcal{L}$ contains unary function symbol f and binary function symbol g, and x and y are variable symbols, then the following are $\mathcal{L}$-terms: fx, gxy, $fgyx$, $gfxy$ and $fgfxfgyz$.*
3. *Among the terms of $\mathcal{L}_A = (0, s, +, \cdot\ ; =)$ are 0, $s0$, $+0ss0$ and $\cdot ss0ss0$. (In the standard interpretation of the symbols, these terms denote the numbers 0, 1, 2, and 4, respectively.)*

A term is called closed or ground if it contains no variable symbols.

No punctuation is required to ensure terms have a unique parsing, but we sometimes use function notation with parentheses, as in $f(x)$ and $g(x, y)$, for readability. For example $fgfxfgyz$ is (arguably) easier to understand when written $f(g(f(x), f(g(y, z))))$. Also, we often use infix notation for binary functions to improve readability. For example, we normally write $t_1 + t_2$ rather than $+t_1t_2$.

Definition 2. The first order formulas for vocabulary $\mathcal{L}$ (or $\mathcal{L}$-formulas) are defined inductively by the following rules.

1. If P is a k-ary relation symbol of $\mathcal{L}$, and $t_1, \ldots t_k$ are $\mathcal{L}$-terms, then $Pt_1 \ldots t_k$ is an $\mathcal{L}$-formula. We call $Pt_1 \ldots t_k$ atomic, because no sub-string of it is a formula.
2. If A and B are $\mathcal{L}$-formulas, then $\neg A$, $(A \wedge B)$, and $(A \vee B)$ are $\mathcal{L}$-formulas.
3. If A is an $\mathcal{L}$-formula, and x a variable symbol, then $\forall xA$ and $\exists xA$ are $\mathcal{L}$-formulas.

The connectives $\wedge$, $\vee$ and $\neg$ are read "and", "or" and "not", as before. The universal quantifier $\forall$ is read "for every", and the existential quantifier $\exists$ is read "for some".

We use the usual abbreviations: $(A \rightarrow B)$ means $(\neg A \vee B)$, and $(A \leftrightarrow B)$ means $((A \rightarrow B) \wedge (B \rightarrow A))$. We also use $t_1 = t_2$ to mean $= t_2 t_2$ and $t_1 \neq t_2$ for $\neg(t_1 = t_2)$.

Example 3.

1. *Formulas of $\mathcal{L}_G$, the vocabulary of graphs, include Exy, Exx, $\forall u \forall v Euv$, $\forall x \exists y Exy$ and $\forall u \forall v_1 \forall v_2 (Euv_1 \wedge Euv_2 \rightarrow v_1 = v_2)$.*
2. *Let $\mathcal{L} = (f, g; P, Q)$, where f is a unary function symbol, g is a binary function symbol, P is a unary predicate symbol, and Q is a binary predicate symbol. Then, the $\mathcal{L}$-formulas include Px, Qxy, Pfx, $(Pfx \wedge Qfxgyx)$, and $\forall x(Px \rightarrow \exists y Qxy)$.*
3. *Formulas of $\mathcal{L}_A$ include $0 = 0$, $s0 \neq 0$, $\forall x \forall y((x + y) = (y + x))$.*

Any sub-string of A that is a formula is a sub-formula of a A. A is a sub-formula of itself; proper sub-formulas are proper sub-strings.

An occurrence of variable x in a formula A is called bound if it is in a sub-formula of A of the form $\forall x B$ or $\exists x B$, and otherwise is free. A formula with no free variables is called a sentence (or ground formula).

2 Semantics of First Order Logic

The semantics of a logic are defined by its satisfaction relation, for which we use the symbol $\models$. For propositional logic, $\models$ is a relation between truth assignments and formulas. This simple semantics is possible because propositional formulas only involve simple propositions. Formulas of first order logic can "talk about" functions, sets and relations, so require semantic objects that contain these. The family of objects in question are called (mathematical) structures, and are simply defined but very general.

A structure $\mathcal{M}$ is a set M (the universe or domain of discourse), together with a collection of functions and relations over M. Structures which might be familiar include graphs, algebraic objects such as groups, and matrices. We use the structure $\mathcal{N}$ of the natural numbers with arithmetic daily. The role of a structure in logical semantics is to assign a meaning, or "interpretation", to each term and each relation symbol.

Definition 3. A structure $\mathcal{M}$ for vocabulary $\mathcal{L}$ (or $\mathcal{L}$-structure) is a tuple consisting of:

1. A non-empty set M, called the universe (or domain) of $\mathcal{M}$;

2. For each k-ary function symbol f of $\mathcal{L}$, a k-ary function $f^{\mathcal{M}} : M^k \to M$;
3. For each k-ary predicate symbol P of $\mathcal{L}$, a k-ary relation $P^{\mathcal{M}} \subset M^k$.

A structure $\mathcal{M}$ for $\mathcal{L}$ is sometimes called an interpretation for $\mathcal{L}$, and if S is a vocabulary symbol of $\mathcal{L}$, then $S^{\mathcal{M}}$ may be called the interpretation of S in $\mathcal{M}$.

If $=$ is included in a vocabulary, it is required to denote true equality (identity of objects). All other predicate symbols may be given arbitrary interpretations. For example, if $<$ is a binary predicate symbol, then $<^{\mathcal{M}}$ can be any binary relation, and need not be an order relation. In practice, we try to choose symbols mnemonically, and normally would use $<$ only for a relation intended to be an order.

Example 4.

1. *Any graph $\mathcal{G} = \langle V, E\rangle$ is a structure for the vocabulary of graphs $\mathcal{L}_{\mathcal{G}} = (E, =)$, although in the notation of these notes we would write it $\mathcal{G} = \langle G, E^{\mathcal{G}}\rangle$, with $G = V$.*
2. *Let $\mathcal{L} = (f; P)$, where f is a unary function symbol and P is a unary relation symbol. One example of an $\mathcal{L}$-structure is $\mathcal{M} = \langle M, P^{\mathcal{M}}, f^{\mathcal{M}}\rangle$, where $M = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, $P^{\mathcal{M}} = \{\mathtt{a}, \mathtt{b}\}$, and $f^{\mathcal{M}} = \{(\mathtt{a} \mapsto \mathtt{b}), (\mathtt{b} \mapsto \mathtt{a}), (\mathtt{c} \mapsto \mathtt{a})\}$.*
3. *The standard structure for $\mathcal{L}_A$ is the structure $\mathcal{N}$ of the natural numbers with:*
   - $N = \mathbb{N} = \{0, 1, 2, \ldots\}$;
   - $0^{\mathcal{N}} = 0$;
   - $s^{\mathcal{N}}$ *is the successor function, i.e., the function that adds one;*
   - $+^{\mathcal{N}}$ *and* $\cdot^{\mathcal{N}}$ *are the standard operations of addition an multiplication;*
   - $=^{\mathcal{N}}$ *is equality.*

We wish to define the satisfaction relation between $\mathcal{L}$-structures and $\mathcal{L}$-formulas. An effect of this definition will be that, for every L-structure $\mathcal{M}$, every sentence for $\mathcal{L}$ will be either true or false in $\mathcal{M}$. We need to define one more term before we can define satisfaction.

An object assignment (sometime called a valuation) for structure $\mathcal{M}$ is a function mapping each variable x_i to an element of the universe M. The purpose of an object assignment is to give a meaning to free variables in formulas. This is needed because we cannot say whether a formula with free variables is true or not, without knowing which objects those variables denote. For example, we do not know if $x + y < 5$ unless we know which numbers x and y are supposed to denote.

We are now in a position to define the semantics of terms, and then of formulas.

Definition 4. The denotation (or meaning) of term t in structure $\mathcal{M}$ with valuation σ, written $t^{\mathcal{M}}[\sigma]$, is defined by recursion on the structure of t, according to the following rules.

1. If t is a variable x, then $t^{\mathcal{M}}[\sigma] = \sigma(x)$;
2. If t is a term of the form $ft_1 \dots t_k$, then $t^{\mathcal{M}}[\sigma] = f^{\mathcal{M}}(t_1^{\mathcal{M}}[\sigma], \dots t_k^{\mathcal{M}}[\sigma])$.

Although notationally a bit heavy, this says the obvious (described operationally): to determine what object $ft_1 \dots t_k$ denotes (according to $\mathcal{M}$ and σ), find what function f denotes (according to $\mathcal{M}$), figure out what objects the arguments $t_1, \dots t_k$ denote (according to $\mathcal{M}$ and σ), and then apply the function to those objects.

Example 5.

1. *Let $\mathcal{L}$ and $\mathcal{M}$ be as in Example 4.2, and σ be such that $\sigma(x) = \mathtt{a}$. Then $(ffx)^{\mathcal{M}}[\sigma] = f^{\mathcal{M}}((fx)^{\mathcal{M}}[\sigma]) = f^{\mathcal{M}}(f^{\mathcal{M}}(x^{\mathcal{M}}[\sigma]) = f^{\mathcal{M}}(f^{\mathcal{M}}(\sigma(x))) = f^{\mathcal{M}}(f^{\mathcal{M}}(\mathtt{a})) = f^{\mathcal{M}}(\mathtt{b}) = \mathtt{a}$.*
2. *$sso^{\mathcal{N}} = s^{\mathcal{N}}(s^{\mathcal{N}}(0^{\mathcal{N}})) = s^{\mathcal{N}}(s^{\mathcal{N}}(0)) = s^{\mathcal{N}}(1) = 2$.*

Notation: If σ is an object assignment, then $\sigma(x \mapsto \mathtt{a})$ denotes the object assignment such that $\sigma(x \mapsto \mathtt{a})(x) = \mathtt{a}$ and $\sigma(x \mapsto \mathtt{a})(y) = \sigma(y)$ when y is a variable other than x.

Definition 5. The satisfaction relation $\mathcal{M} \models A[\sigma]$ is defined by recursion on the structure of A, according to the following rules.

1. $\mathcal{M} \models Pt_1 \dots t_k\ [\sigma]$ iff $\langle t_1^{\mathcal{M}}[\sigma], \dots t_k^{\mathcal{M}}[\sigma] \rangle \in P^{\mathcal{M}}$
2. $\mathcal{M} \models t_1 = t_2\ [\sigma]$ iff $t_1^{\mathcal{M}}[\sigma] = t_k^{\mathcal{M}}[\sigma]$
3. $\mathcal{M} \models \neg B\ [\sigma]$ iff $\mathcal{M} \not\models B\ [\sigma]$
4. $\mathcal{M} \models (B \vee C)\ [\sigma]$ iff $\mathcal{M} \models B\ [\sigma]$ or $\mathcal{M} \models C\ [\sigma]$;
5. $\mathcal{M} \models (B \wedge C)\ [\sigma]$ iff $\mathcal{M} \models B\ [\sigma]$ and $\mathcal{M} \models C\ [\sigma]$;
6. $\mathcal{M} \models \forall x B\ [\sigma]$ iff for every $\mathtt{a} \in M$, $\mathcal{M} \models B\ [\sigma(x \mapsto \mathtt{a})]$
7. $\mathcal{M} \models \exists x B\ [\sigma]$ iff for some $\mathtt{a} \in M$, $\mathcal{M} \models B\ [\sigma(x \mapsto \mathtt{a})]$

Notice that item 2 in Definition 5 follows from item 1 and the requirement that $=$ always denotes equality.

Example 6.

1. $\mathcal{N} \models (x + y) = sssss0\ [\sigma]$ *iff* σ *is such that* $\sigma(x) + \sigma(y) = 5$.
2. $\mathcal{N} \models \forall x \exists y (y = ssx)$, *regardless what* σ *is, because the formula has no free variables, and for every natural number there is another that is larger by two.*
3. $\mathcal{N} \not\models \forall x \exists y (ssy = x)$, *regardless what* σ *is, because the formula has no free variables, and there is no natural number two smaller than 1 or 0.*
4. *Let* $\mathcal{M}$ *be an* $\mathcal{L}_A$*-structure with* $M = \{0, 1, 2\}$, $0^{\mathcal{M}} = 0$, $s^{\mathcal{M}}(x) = x + 1 \bmod 3$, *and* $+^{\mathcal{M}}$ *be addition modulo* 3. *Then* $\mathcal{M} \models (x + y) = sssss0\ [\sigma]$ *iff* σ *is such that* $\sigma(x) + \sigma(y) = 2$ *mod* 3.
5. *Let* $\mathcal{M}$ *be an* $\mathcal{L}_G$*-structure with* $M = \{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ *and* $E^{\mathcal{M}} = \{(\mathtt{a}, \mathtt{b}), (\mathtt{b}, \mathtt{c})\}$. *Then* $\mathcal{M} \models Euv[\sigma]$ *if* $\sigma(u) = \mathtt{a}$ *and* $\sigma(v) = \mathtt{b}$, *but* $\mathcal{M} \not\models Euv[\sigma]$ *if* $\sigma(u) = \mathtt{a}$ *and* $\sigma(v) = \mathtt{c}$
6. *Let* $\mathcal{G}$ *be a graph. Then* $\mathcal{G} \models \forall u \forall v Euv$ *if and only if* $\mathcal{G}$ *is a complete graph.*

If A is a sentence, then σ plays no role in the truth of A: For each structure $\mathcal{M}$ (for the same vocabulary as A), either $\mathcal{M} \models A\ [\sigma]$ for every σ or for no σ. Thus, we may leave out σ and simply write $\mathcal{M} \models A$, or $\mathcal{M} \not\models A$. In the case that $\mathcal{M} \models A$, we say that "A is true in $\mathcal{M}$", or that "$\mathcal{M}$ satisfies A", or that "$\mathcal{M}$ is a model for A".

The size of structure $\mathcal{M}$ is the size of its universe M, and we call $\mathcal{M}$ finite if M is finite.

Definition 6. The problem of model checking for first order logic (FO) on a class C of finite structures is:

> Given: A (string encoding a) structure $\mathcal{M}$ from C and a FO sentence A;
> Question: Does $\mathcal{M} \models A$?

Fact 1.

1. *If* C *is all structures, FO model checking for* C *is undecidable.*
2. *If* C *is all finite structures, FO model checking for* C *is PSPACE complete.*
3. *If* C *contains only finite structures, FO model checking for* C *can be done in time* $O(n^k)$, *where* n *is the size of the structure, and* k *the size of the sentence (in symbols, counting all terms).*
4. *If* C *contains* $\mathcal{N}$, *the structure of the natural numbers with addition and multiplication, then FO model checking for* C *is undecidable.*

To see that item 3 is true, observe that a recursive algorithm directly implementing the recursion of Defintions 5 and 4 runs in the claimed time.

3 Satisfiability, Validity, Equivalence and Logical Consequence

Definition 7.

1. Formula A is satisfiable iff $\mathcal{M} \models A[\sigma]$, for some $\mathcal{M}$ and σ, and otherwise is unsatisfiable;
2. Set Φ of formulas is satisfiable iff there is some $\mathcal{M}$ and σ, such that $\mathcal{M} \models A[\sigma]$ for every $A \in \Phi$;
3. Formula A is valid (written $\models A$), iff $\mathcal{M} \models A[\sigma]$ for every $\mathcal{M}$ (of suitable vocabulary) and every σ;

Fact 2.

1. *The problem "given a FO formula A, is A satisfiable?", is undecidable.*
2. *The problem "given a FO formula A, is A valid", is undecidable.*

One might hope the the undecidability is a consequence of the fact that some formulas are satisfiable but have only infinite models. But this is not the case.

Definition 8. Formula A is finitely satisfiable iff $\mathcal{M} \models A\ [\sigma]$, for some finite $\mathcal{M}$ and σ.

Fact 3.

1. *The problem "given a FO formula A, is A finitely satisfiable?", is undecidable.*
2. *The problem "given a FO formula A and natural number k, does A have a model of size k?" can be decided in time $O(2^{(k^n)})$, where n is the size of the formula.*

Definition 9.

1. A is a logical consequence of Φ (written $\Phi \models A$) iff for every $\mathcal{M}$ and σ, $\mathcal{M} \models A\ [\sigma]$;
2. A and B are logically equivalent (written $A \equiv B$) iff, for all $\mathcal{M}$ and σ, $\mathcal{M} \models A\ [\sigma]$ iff $\mathcal{M} \models B\ [\sigma]$.

Example 7.

1. *Does* $(\forall xA \vee \forall xB) \models \forall x(A \vee B)$ *hold for every two formulas A,B? Yes. Suppose* $\mathcal{M}$ *and* σ *are such that* $\mathcal{M} \models (\forall xA \vee \forall xB)[\sigma]$. *Then, by Definition 5, either* $\mathcal{M} \models \forall xA[\sigma]$ *or* $\mathcal{M} \models \forall xB[\sigma]$. *Suppose the first case. Then, for every* $a \in M$, $\mathcal{M} \models A[\sigma(a/x)]$, *so, for every* $a \in M$, $\mathcal{M} \models (A \vee B)[\sigma(a/x)]$. *Therefore* $\mathcal{M} \models \forall x(A \vee B)[\sigma]$. *The symmetric argument applies if* $\mathcal{M} \not\models \forall xA[\sigma]$, *but* $\mathcal{M} \models \forall xB[\sigma]$.

2. *Does* $\forall x(A \vee B) \models (\forall xA \vee \forall xB)$ *hold for every two formulas A,B? No. Let A be* Px, *and B be* Qx, *and let* $\mathcal{M}$ *be the structure with* $M = \mathbb{N}$, $P^{\mathcal{M}}$ *the set of even natural numbers, and* $Q^{\mathcal{M}}$ *the set of odd natural numbers. Then* $\mathcal{M} \models \forall x(A \vee B)$, *because every natural number is either even or odd, but* $\mathcal{M} \not\models (\forall xA \vee \forall xB)$, *because some natural numbers are not even, and some are not odd.*

Exercise 1. *Verify each of the following.*

1. *For every formula* A, $\neg\forall xA \equiv \exists x\neg A$.
2. *For every formula* A, $\neg\exists xA \equiv \forall x\neg A$.
3. *For every two formulas* A, B, $(\forall xA \wedge \forall xB) \equiv \forall x(A \wedge B)$.
4. *For every two formulas* A, B, $(\exists xA \vee \exists xB) \equiv \exists x(A \vee B)$.
5. *For every two formulas* A, B, $\exists x(A \wedge B) \models (\exists xA \wedge \exists xB)$.
6. *There are formulas* A, B *for which* $(\exists xA \wedge \exists xB) \not\models \exists x(A \wedge B)$.
7. *For every formula* A, $\forall x\forall yA \equiv \forall y\forall xA$.
8. *For every formula* A, $\exists x\forall yA \models \forall y\exists xA$.
9. *There are formulas* A *for which* $\forall x\exists yA \not\models \exists y\forall xA$.
10. *For every formula* A, $\forall xA \models \exists xA$.

Fact 4.

1. *The problem "given two FO formulas* A, B, *does* $A \models B$", *is undecidable, as is checking if* $A \equiv B$.

2. *The problem "given two FO formulas* A, B, *does A finitely entail B (that is, is every finite model of A also a model of B), is undecidable, as is checking if A and B have the same finite models.*